Evaluating latent representations in encoder-decoder models for text summarization with multitask and transfer learning.

Justin J. Grace

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2018

Abstract

Neural encoder-decoder models have shown significant promise in sequence transduction tasks such as machine translation and, more recently, text summarization. Such models rely on a latent space representation of the source sequences, learned by the encoder, to pass to a decoder for generation of a target sequence. Understanding and evaluating what information the latent representations learn remains a significant challenge. Multitask learning aims to solve several tasks simultaneously using learned representations that are shared across different tasks. Our research investigates learned latent space representations of the encoder in the context of different sequence transduction tasks and multitask learning. We make use of a novel dataset of news articles from the Guardian newspaper, which are accompanied by metadata including short summaries and topic tag sequences for articles. We train separate encoder-decoder recurrent neural network models to generate (a) abstractive text summaries of the articles and (b) topic tag sequences related to the article content. We first establish high quality benchmarks in the new dataset and on the new task. We then perform experiments using our models to manipulate the latent representations learned by the models using multitask learning. We train an encoder-dual-decoder model to perform both summarization and tag sequence generation simultaneously. Whilst performance of the single task models is good the multitask model fails to learn to generate high quality sequences. We evaluate learned representations using transfer learning with a semantic classification task. We show that the tag sequence generation model learns representations that are more useful for the semantic classification side task and by training a summarization model with a multitask objective we induce a similar performance increase on the side task.

Acknowledgements

I would like to thank my two supervisors, Shay Cohen and Ivan Titov, for all the advice and flexibility they provided during my project. I feel privileged to have had access to such brilliant people.

Thanks to my family, Sue, Andrew and Theo, for being infinitely supportive during the Masters. I would also like to thank my friends, both old and new, who have made the year joyful and the stress more bearable.

Finally, I would like to dedicate this work to Ahmed Mustafa, a friend and fellow student who was tragically taken from us this year. A brilliant mind and hilarious, kind character who will be missed.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

[illegible]

(*Justin J. Grace*)

Table of Contents

Chapter 1

Introduction

1.1 Motivation

The advent of deep neural network approaches to machine learning, and their apparent success, has led to a paradigm shift across many domains of computer science research, and in several cases dramatically increased the state-of-the-art, including speech recognition, visual object recognition, object detection and language modelling (LeCun et al., 2015). However, these new methods come with a new set of challenges that require understanding if we are to effectively make use of them for real-world tasks.

A key aspect of machine learning is the process of representation learning; the process of transforming source data into an alternative form of features that is more useful in the context of a given task. Non-neural methods typically involve hand-engineering of these features, which makes them explicit. Neural methods, however, learn these feature representations implicitly; dependent on an explicitly specified architecture. This creates a trade-off between increased expressibility but decreased interpretability (LeCun et al., 2015): it is less clear what information in the source data the learned representations reflect. This has led to deep neural network (DNN) methods being described as ”black boxes” (Shwartz-Ziv and Tishby, 2017) and to a new area of research around understanding representation learning and improving interpretability.

This research investigates representation learning in the context of abstractive summarization using a novel dataset. The dataset is a collection of news articles from the Guardian newspaper [1] with summaries and topic information. Abstractive summarization refers to the task of generating a shorter summary from a longer source text

[1]www.theguardian.com

without relying on explicitly copying entire phrases from the source text (which is known as extractive summarization) (Rush et al., 2015). Abstractive summarization requires a model that can generate sequences of fluent and adequate text, which is considered more challenging than identifying text to be extracted since it additionally requires information about how to generate natural language (Rush et al., 2015). A key question this research investigates is whether abstractive summarization also learns information regarding the global semantics of a source text, in particular regarding the abstract semantic class of a whole document. To investigate this question we present a side task as a metric for semantic information contained in the learned representations by evaluating their capacity for transfer learning (TL). Each article appears within a section of the newspaper, such as politics, fashion or football, which we use as a global semantic category label.

We introduce a novel application of the encoder-decoder model to generate topic tag sequences that are metadata associated with each document. These are not natural language but we treat this as a sequence generation task as an experiment to explore whether these models can also handle more general sequences. Finally we train a multitask learning (MTL) model that is capable of both summary generation and topic tag sequence generation (TTSG) simultaneously. We hypothesize that the auxiliary TTSG task will encourage the model to learn more semantically aligned representations that also lead to higher quality summaries.

We evaluate global semantic information of the latent representations by training a classifier, using TL, on top of intermediary layers in the models to investigate whether the learned representations retain information about the global semantic class of the source text. We use this novel evaluation tool to compare learned representations across our models for single-task summarization, single-task TTSG and MTL models for join summarization and TTSG.

1.2 Objectives

This work has five objectives: to describe a novel dataset in the context of encoder-decoder architectures for sequence transduction; to create baseline models for the task of abstractive summarisation with the new dataset; to provide some insights into learned text sequence representations by analyzing the outputs of the encoder and using an auxiliary semantic classification task to measure semantic information of the representations; to investigate a novel application of encoder-decoder models for generating topic tag sequences; and to investigate whether MTL on abstractive summarization and TTSG improves performance on either or both tasks, and whether the MTL encourages the encoder to learn more global semantic information (as measured by our novel auxiliary TL semantic task).

1.3 Summary Results

We are able to train good benchmark models using the novel dataset and achieve ROUGE-1 scores of 35.94 with a Pointer-Generator Network (a recent state-of-the-art, SOTA, model; See et al. (2017), and 26.60 with our own Encoder-Decoder with attention model (model similar to Luong et al. (2015a)), which is an improvement over the same model architecture trained using Open-NMT (Klein et al., 2017).

This model was also reasonably capable of learning to generate the topic tag sequences, scoring a F1 score of 58.73 (Open-NMT model, 140k steps) and 42.49 (our model, 15k steps). Our model outperformed the Open-NMT model at equivalent training steps, however our model was slow to train and we were unable to extend training to match the Open-NMT model.

Despite the relative success of the two single-task models, training by MTL led to catastrophic failure, and the model was unable to generate good sequences for either task. The model appears to overfit high-frequency tokens leading to repetition behaviour maximizing precision but leading to poor recall and inadequate, non-fluent outputs.

For the semantic side task we demonstrate a performance range between a lower bound of 12.4% (chance level) and 23.75% (accuracy achieved by a fully-trained task-specific model). We show that the summarization single-task model learns some information useful for the TL semantic task but less than is learned by the TTSG model or the MTL models. Despite the inadequacy of the MTL outputs, the auxiliary task does seem to have behaved as expected and imbued the latent representations with more semantic information.

1.4 Document Structure

The structure of this document is as follows: chapter 1 (this chapter) provides an introduction to the work, outlining the motivation, research questions and objectives, and a brief summary of results, chapter 2 contains background literature for this research, chapter 3 describes the dataset, chapter 4 details the methods and implementation for the research, chapter 5 reports the findings, our analysis and relevant discussion and chapter 6 draws conclusions and considers implications for future work.

Chapter 2

Background

2.1 Neural Networks

Deep neural networks are now the status quo for many machine learning problems. A single layer neural network (also known as a perceptron) is a combination of affine transformations, whose parameters are learned by gradient descent by backpropagation of errors, and a non-linear activation function (Rumelhart et al., 1986). Multiple networks can be stacked on top of each other, where the outputs of one layer become the inputs of another, to form [in this case] a multilayer perceptron (MLP, equations 2.1 & 2.2, figure 2.1).

$$\mathbf{y} = g(\mathbf{W}^{(1)}\mathbf{h} + \mathbf{b}^{(1)}) \tag{2.1}$$

$$\mathbf{h} = f(\mathbf{W}^{(2)}\mathbf{x} + \mathbf{b}^{(2)}) \tag{2.2}$$

Where f is the sigmoid function and g is the softmax function.

MLPs have been shown to be universal function approximators (Hornik et al., 1989); that is, they have the capacity to represent any mathematical function [1]. This is a mixed blessing since it means that the function may have very high complexity whilst also being learned implicitly; in other words it might present with very high performance but without any explicitly interpretable explanation as to how it does so.

[1]This does not mean that they *will* learn any function, only that they theoretically have the capacity to. The learned function is still constrained in practice by the data, architecture and hyperparameters

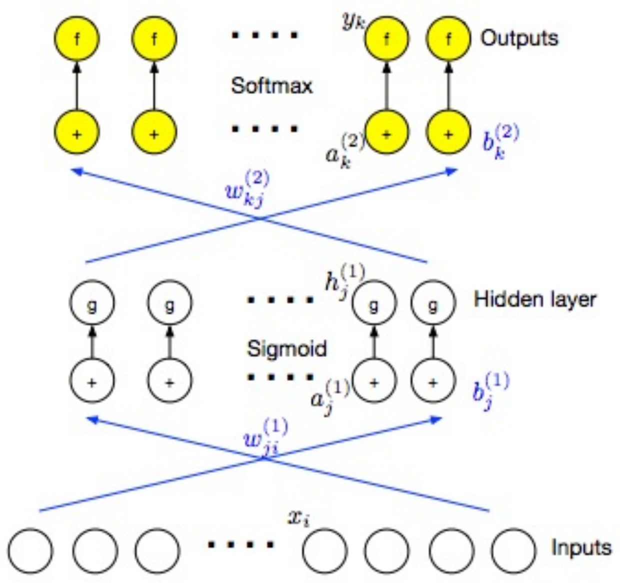

Figure 2.1: Diagram of a 2 layer perceptron. From Renals (2017).

2.2 Representation Learning

Neural networks trained by supervised learning are considered to perform a kind of representation learning (Goodfellow et al., 2016). For example, if we consider a model where the final layer is a softmax regression classifier, the ideal representation for the penultimate layer would be one where all classes are linearly separable.

Figure 2.2 shows a two dimensional approximation[2] of MNIST handwritten digits based on their vectorized pixel representation whilst figure 2.3 shows the same digits using the latent space (i.e. hidden layer outputs) from a neural network classifier model. It is clear that the representations learned by the model are more suitable for the task of classifying digits since the clusters of matching digits are denser and more coherent.

For more complex models it is not always clear what the ideal learned representation is. Bengio et al. (2012) provide suggestions for qualities (or ”priors”) that good representations should have, including *natural clustering*; the idea that different classes of entities should form natural clusters in the latent space, as shown in figure 2.3. Our research approaches this question in the context of encoder-decoder models for text generation (detailed later in this chapter) by probing whether the encoder learns information relevant for other linguistic tasks, such as global semantic classification (pre-

[2]using t-distributed Stochastic Neighborhood Embedding, or t-SNE.

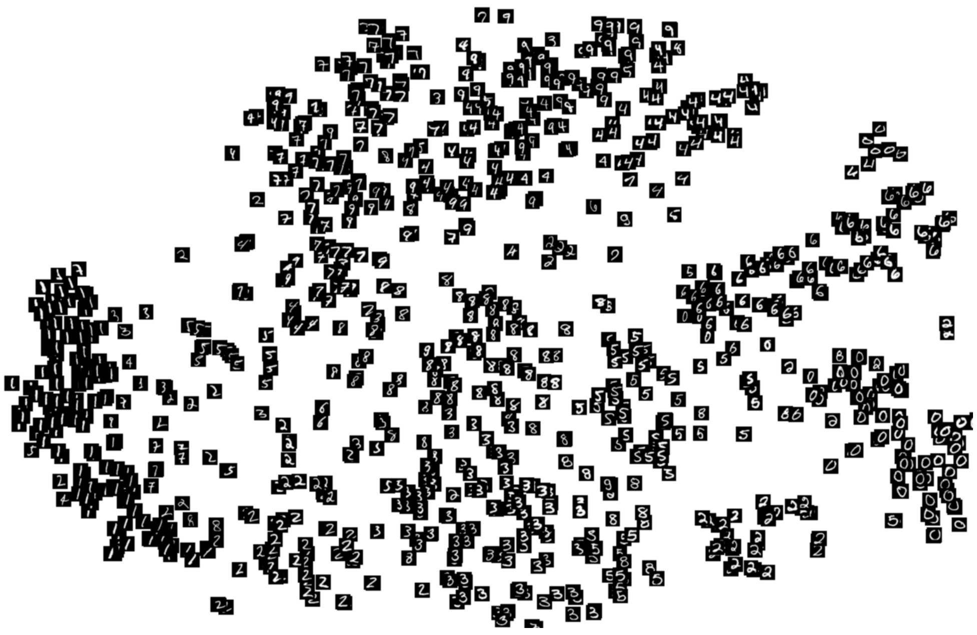

Figure 2.2: A 2 dimensional projection of the image space for MNIST digits. From Despois (2017).

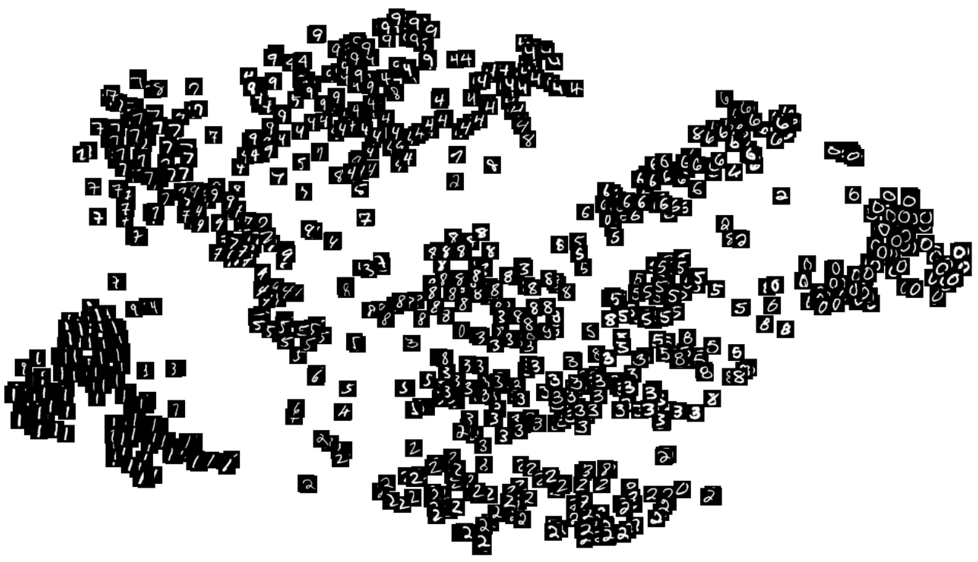

Figure 2.3: A 2 dimensional projection of the latent space for MNIST digits. From Despois (2017).

dicting whether a text is about football or politics), which, intuitively, seems important for the task of abstractive summarization. Alternatively, the model may be learning a complex copying function that simply relies on positional information and therefore learn nothing about the semantics of the document.

2.3 Word Embeddings

Word embeddings are a method for converting discrete text into real-valued vectors that capture linguistic regularities (Bengio et al., 2003). They typically lead to significant improvements when training DNNs on natural language (Luong et al., 2013). They are also usually formed from the latent space representation of a model (although not necessarily deep or neural). For example, one method, known as skip-gram (Mikolov et al., 2013b) iterates through the text in a document word-by-word and gathers the surrounding context words, $C = \{c_{-n}, ..., c_{-1}, c_{+1}, ..., c_{+n}\}$, to either side of this pivot word, w_0. A model is then trained to predict the context words, C, one by one, given the pivot word, w_0. In practice this model is often approximated by training a classifier over a set of candidate context words, C' using negative sampling (Mikolov et al., 2013a). Once the model is trained, the last layer of the model which predict the context word (or discriminates between candidate context words) can be discarded since we are only interested in the mapping function from a word as input to the real valued vector in the latent space of the model. Figure 2.4 shows a two dimensional approximation (using t-SNE) of the high dimensional vector space learned by training skip-gram on Harry Potter novels; it is clear that distributionally related words appear closer in the embedding space.

There are a number of methods for generating useful latent space representations of words. In this research we use pre-trained GloVe representations (Pennington et al., 2014). GloVe representations are similar to skip-gram representations but they directly consider word occurrence statistics (whilst skip-gram is only exposed to this information implicitly). The GloVe algorithm constructs a co-occurrence matrix of words, which is then factorized to reduce it's dimensionality.

2.4 Natural Language & Recurrent Neural Networks

A defining attribute of natural language is that units of meaning, be they words, phrases, sentences, or paragraphs can be of arbitrary length. For some time this made language data difficult to work with using DNNs since most of the architectures required inputs of fixed size. Recurrent neural networks (RNN) are a family of architectures that are capable of encoding variable length inputs. With RNNs we define a scale of interest

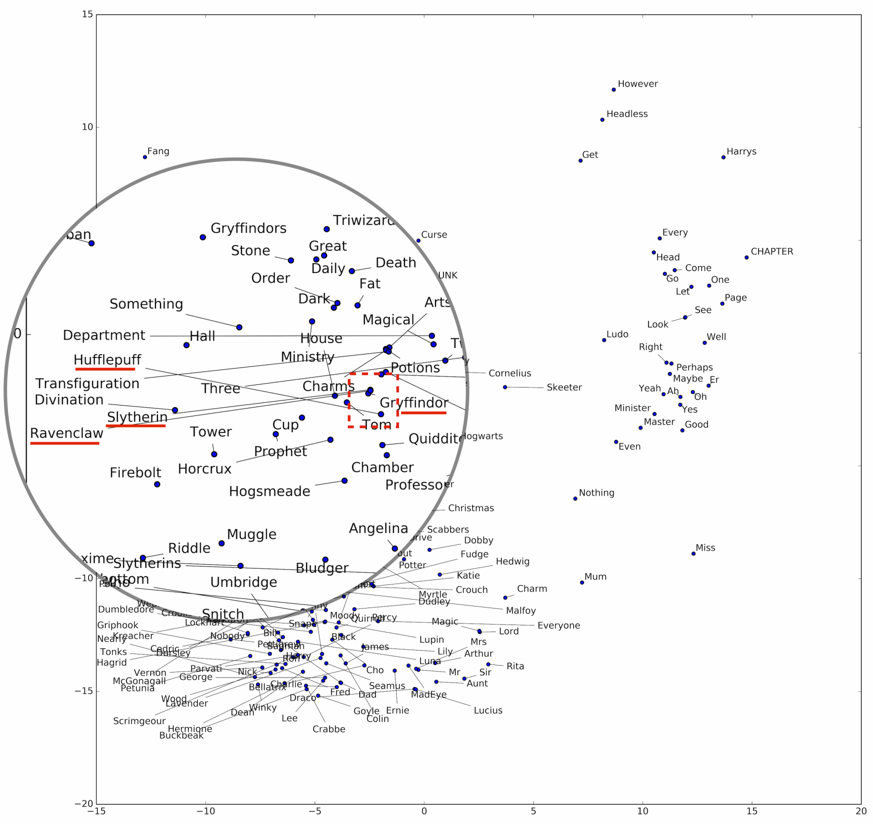

Figure 2.4: Word embeddings trained by skip-gram on all 7 Harry Potter books. From Despois (2017).

(e.g. characters, words, phrases) and we step through the input data item by item. Whilst other model architectures only consider the immediate inputs connected to the network at a given timestep, RNNs have a memory, they preserve the hidden representation from the previous timestep, h_{t-1} and include this with the hidden representation of the current timestep, h_t (e.g. by adding the two vectors; see equations 2.3 and 2.4 and figure 2.5).

$$\mathbf{h}_t = tanh(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{xh}\mathbf{x}_t) \tag{2.3}$$

$$\mathbf{y} = g(\mathbf{W}_{yh}\mathbf{h}_t) \tag{2.4}$$

where $\mathbf{y}$ are the outputs of the RNN, $\mathbf{h}_t$ are the hidden states at the current timestep,

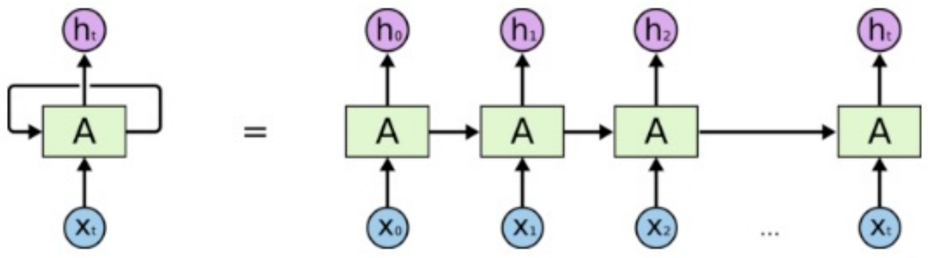

Figure 2.5: Diagram of rolled and unrolled RNN. From Olah (2017).

$\mathbf{x}_t$ are the inputs at time t, $\mathbf{W}_{hh}, \mathbf{W}_{xh}, \mathbf{W}_{yh}$ are initialized weight matrices, and $\mathbf{h}_{t-1}$ is the hidden state from the previous timestep. g is a non-linear function that is appropriate for the desired output, for example, a language model might predict the next word in the sequence, y, where $x_1,...,x_t$ are the preceding words, by estimating the probabilities of words in a fixed size vocabulary, V, with g as a softmax:

$$p(y) = \prod_{t=1}^{T} p(y_j | x_t,...,x_1) \tag{2.5}$$

$$p(y_j = 1 | x_t,...,x_1) = \frac{exp(\mathbf{w}_j \mathbf{h}_t)}{\sum_{j'=1}^{K} exp(\mathbf{w}_{j'} \mathbf{h}_t)} \tag{2.6}$$

for all possible words, $j = 1,...,K$, from the vocabulary, V, where $\mathbf{w}_j$ are the rows of the weight matrix from equation 2.4, $\mathbf{W}_{yh}$.

Whilst RNNs like this solved the variable length input problem they were shown to have limitations in practice, in particular, they were prone to vanishing gradients leading to an inability to preserve signals from inputs many timesteps prior; effectively truncating the memory (Bengio et al., 1994). Long Short-term Memory (LSTM) networks were developed to address this issue, by adding a number of learned soft gating functions to the RNN they are able to preserve long-range signal dependencies when appropriate (Hochreiter and Schmidhuber, 1997). LSTMs and other similar architectures enable the use of long input sequences such as multiple sentences or even whole documents. This is important for our research since we are training models to respond to entire documents as inputs.

2.5 Sequence Generation & Encoder-Decoder Models

Tasks such as document summarization and language translation require models that can not only consider variable length inputs but also generate variable length outputs. Cho et al. (2014) proposed the RNN encoder-decoder model[3] which uses a RNN to take inputs of arbitrary length and encode these to a fixed size vector representation, and a second RNN that takes this latent representation as an input and generates an output of variable length, determined by the prediction of a terminal symbol. They demonstrate this model in the context of machine translation and demonstrate, qualitatively, that the model learns a semantically and syntactically meaningful representation of linguistic phrases (Cho et al., 2014).

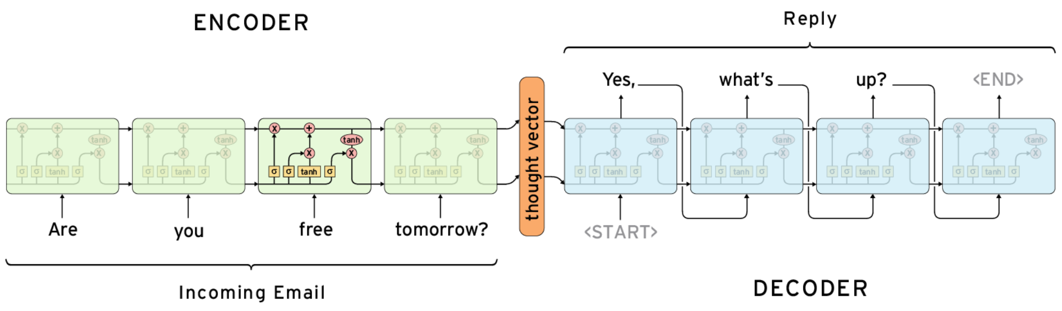

Figure 2.6: Diagram LSTM Encoder-Decoder model for generating e-mail replies. Here the output of the encoder is referred to as a 'thought vector'. From Britz (2016).

The loss of an encoder-decoder model is calculated over a target sequence. The encoder RNN is as described in the previous section. The decoder RNN differs in that it is conditioned on more information. It considers an input, c, which is the output of the encoder, the last state of the RNN after forward propagation through the whole input sequence, the previous hidden state of itself, $h_{t-1}^{(dec)}$, but also the previous output of the decoder, $t-1$. The hidden state of the decoder at time t is given by,

$$\mathbf{h}_t^{(dec)} = f(\mathbf{h}_{t-1}^{(dec)}, y_{t-1}, \mathbf{c}) \tag{2.7}$$

Whilst the conditional probability of the next symbol is

$$P(y_t|y_{t-1}, y_{t-2}, ..., y_1, \mathbf{c}) = g(\mathbf{h}_t^{(dec)}, y_{t-1}, \mathbf{c}) \tag{2.8}$$

[3]a similar model was proposed by Sutskever et al. (2014) at the same time, known as sequence2sequence or seq2seq

The encoder and decoder are jointly trained to maximize the conditional log-likelihood

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} \log p_{\theta}(y_n | x_n) \tag{2.9}$$

2.6 Attention Mechanisms & Interpretability

Bahdanau et al. (2014) identified that the fixed-length encoder output vector used as the input to the decoder acts as a bottleneck and proposed an Attention mechanism that enables the model to learn a soft-search function allowing it to selectively attend to parts of the source sequence. They show that, for the task of machine translation, that this function seems to learn alignment between source and target sequences (Bahdanau et al., 2014). Alignment is the process of associating a single output symbol with one (or sometimes more) input symbol when performing sequence transduction. It learns alignment in the case of translation since this information is highly relevant for the task, however it may learn other functions for different tasks (figure 2.7 is visualisation of attention weights, that show some characteristics of alignment) (Ghader and Monz, 2017). Bahdanau et al. (2014) report a new state-of-the-art score for machine translation as measured by BLEU score[4] and, in particular, improved performance on translating long sequences.

Ghader and Monz (2017) study in the function of attention in machine translation in more detail, concluding that for nouns, attention tends to learn alignment but for verbs, attention is often more diffuse, referring to other words outside of the scope of alignment. A number of different implementations for attention have since arisen, in particular, Luong et al. (2015a) investigate several different flavours of attention and their efficacy. They investigate *global* and *local* variants of attention that consider either all source words or a subset of source words at a time, respectively. They show that attention mechanisms add about 5.0 BLEU points over non-attentional state-of-the-art (SOTA) systems. They use an ensemble of their proposed methods to produce a new SOTA result for English to German translation outperforming previous approaches by over 1.0 BLEU.

A useful side affect of attention is that it naturally lends itself to visualisation and lets us peek at the inner workings of DNN to understand what regions of an input the model considers important for the task at hand. As we have seen already, this

[4]BLEU is a precision-centric metric often used to evaluate machine translation and sequence transduction tasks, e.g. see Bahdanau et al. (2014).

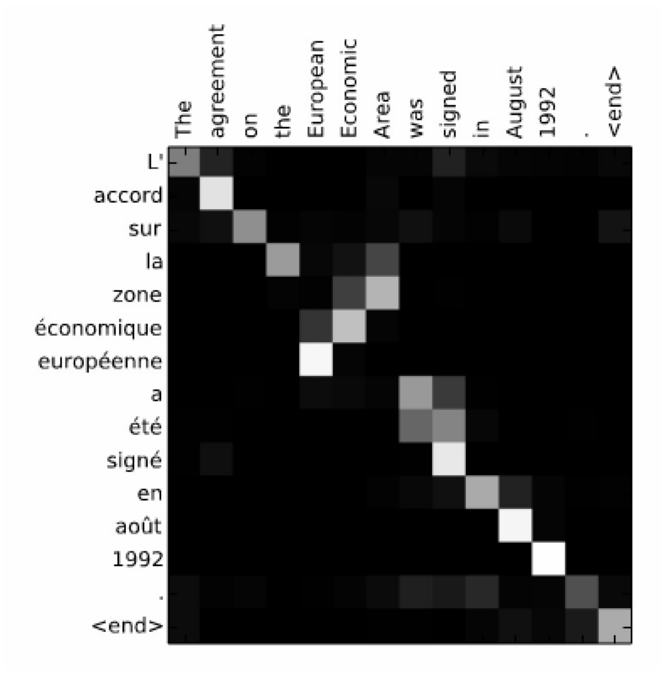

Figure 2.7: Attention matrix showing a learned soft alignment function for machine translation. From Bahdanau et al. (2014).

has been useful for establishing that for NMT, attention leans alignment-like functions with some variation, particularly when considering verbs.

2.7 Text Summarisation Models

The aim of text summarization is to produce a condensed representation of an input text that captures the core meaning of the original (Rush et al., 2015). One approach to this problem which has had good success is known as *extractive* summarization, in which a model learns to copy phrases from the original source text (e.g. Narayan et al. (2017)). *Abstractive* summarization, on the other hand, attempts to produce a bottom-up summary, which may contain novel content not found in the original source. High quality summarization includes sophisticated techniques such as paraphrasing, generalization, or incorporating real-world knowledge that cannot be attained via an extractive only approach (See et al., 2017). Rush et al. (2015) approached the problem of abstractive summarization taking a data-driven approach inspired by the success of attentional encoder-decoder models such as Bahdanau et al. (2014), as described in the previous section. This approach, Attention-based summarization (see figure 2.8), forms the backbone of our own research, and is discussed in detail in the methods section in chapter 4. Their model also includes several other features that improve performance including a beam-search decoder as well as features to facilitate conditional extraction functions.

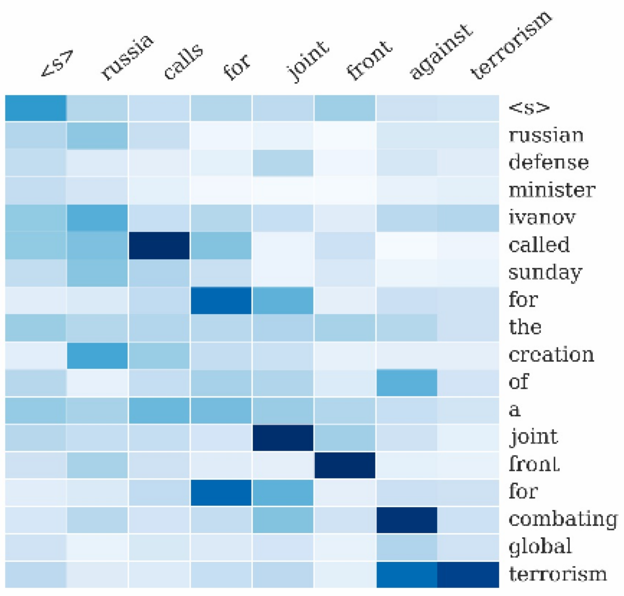

Figure 2.8: A learned attention matrix for abstractive text summarization between the source text (right) and the generated summary (top). It includes examples of copying, paraphrasing with a synecdoche, and grammatical reframing behaviors. From Rush et al. (2015).

Further improvement to the SOTA was made by See et al. (2017) with Pointer-Generator networks. They noticed shortcomings with other neural abstractive systems when having to reproduce facts from the source text and also that they tended to be repetitive. They contributed two innovations; firstly a method for learning to copy via pointing and secondly, a method called coverage, which keeps track of what has already been summarized, discouraging repetition (See et al., 2017). Put simply, the pointer-generator component involves learning an additional generation probability; the probability that the model should generate versus copy. The final probability for a word, w, if given by

$$P_{final}(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i \tag{2.10}$$

in other words, the final probability is the sum of the probability under the language model and the probability under the attention model over the whole source sentence.

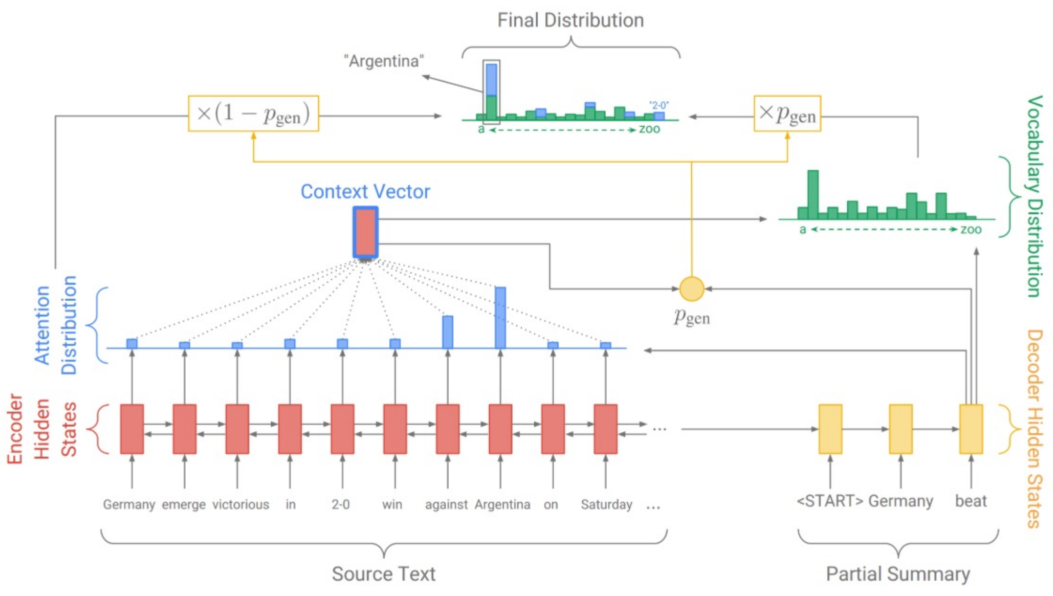

Figure 2.9: A pointer-generator model. For each decoder timestep a generation probability $p_{gen} \in [0,1]$ is calculated, which weights the probability of generating from the vocabulary versus copying from the source. The blue represents contribution from attentional copy model and the green from the generative model. From See et al. (2017)

Coverage is modeled as an extra term in the attention computation that is the sum of all previous attention weights for the current decoder sequence. This ensures that the attention mechanism is aware of previous attention states and discourages repeatedly attending to the same locations. They also define a new loss function, the coverage loss, which penalizes repeatedly attending to the same location. This loss as added to the primary loss function to create a composite loss that serves to optimize over transduction accuracy and coverage.

2.7.1 Evaluating Summarization

Evaluating summarization effectively remains an open challenge (Novikova et al., 2017). There are two main approaches; human and automatic evaluation. Human evaluation is still considered the gold standard but is very resource intensive. Automatic approaches alone are frequently reported in the literature. Most methods revolve around measuring relevance, and do so by computing either the precision or recall of an output relative to the reference sequence. In the context of sequence generation of tokens, precision, P, is ratio of the relevant generated tokens (i.e. generated tokens found in the reference), $\{tok_{rel}\} \cap \{tok_{gen}\}$, to all generated tokens, $\{tok_{gen}\}$

$$P = \frac{|\{tok_{rel}\} \cap \{tok_{gen}\}|}{|\{tok_{gen}\}|} \tag{2.11}$$

whilst recall is the ratio of relevant generated tokens, $\{tok_{rel}\} \cap \{tok_{gen}\}$, to all relevant tokens, $\{tok_{rel}\}$

$$P = \frac{|\{tok_{rel}\} \cap \{tok_{gen}\}|}{|\{tok_{rel}\}|} \tag{2.12}$$

Precision and recall have been modified for the specific case of sequence generation to create BLEU and ROUGE metrics, respectively, which extend precision and recall to account for maximum number of reference token appearances, addressing the issue of repetition, and for longer n-gram sequences (Papineni et al., 2002; Lin, 2004). Machine translation relies heavily on the BLEU score, which is a precision-centric metric. Whilst precision is important in summarization recall is typically considered to be of greater importance.

2.8 Multitask Training in NLP

Collobert and Weston (2008) showed that it was possible to solve a host of NLP tasks using a single (albeit complex) model by framing all tasks as different forms of sequential word-level classification (the tasks include part-of-speech tagging, chunking, named entity recognition, semantic role labeling, language modelling, and synonym recognition). A few aspects of the model are different to the models we have considered so far; they use 1D convolutions rather than RNN and they take a maxpool over time of the encoder states to form the latent representation (as opposed to simply selecting the last state). Perhaps their most interesting finding (from our perspective) is that the joint training across multiple tasks appears to confer performance improvements across all tasks. They speculate that this is due to the learned representations being higher quality from learning more complete information regarding linguistic regularities (Collobert and Weston, 2008). This finding is not in isolation, Liu et al. (2015) show that multitask training leads to improved information retrieval; Pasunuru et al. (2017) use multitask training on abstractive summarization with entailment generation to improve quality; along similar lines Guo et al. (2018) improve abstractive summarization with both entailment and question generation tasks. Ruder (2017) provides a helpful overview of the multitask learning (MTL) paradigm and recently Salesforce

Research released the Natural Language Decathalon (McCann et al., 2018) a challenge benchmark for MTL models to perform ten (existing benchmark) NLP tasks, along with their own model setting an impressive benchmark beating the single-task SOTA on several of the tasks.

It is not definitively clear as to why MTL leads to improved global performance however the accepted wisdom is that the shared representations allow better generalization on tasks, discouraging overfitting and acting as a form of regularization (McCann et al., 2018). There are different methods for performing MTL, for instance *hard* and *soft* parameter sharing, where tasks share layers of the model or where tasks have separate models but the distance of the parameters across models is regularized, encouraging similar representations. Our research focuses on hard parameter sharing.

2.9 Transfer learning in NLP

Transfer learning refers to the practice of pre-training a model using some task other than it's final intended task followed by a fine-tuning process on the intended task (Bengio, 2012). This approach has led to significant performance gains over models trained only on the primary task (Howard and Ruder, 2018). Typically transfer learning is used when there is a poverty of task-specific data but there is some within domain task with a dearth of similar data. Pre-training on the related task can lead to improvements on the primary task (Pan et al., 2010). For example, Johnson et al. (2016) make use of transfer learning to perform ”zero shot” language translation by training cross-lingual embeddings. In our work we use transfer learning as a method for evaluating the learned representations of our models to see how well they generalize to other related tasks.

Chapter 3

Data

3.1 Datasets & Tasks

The Guardian news article data were collected from their public data API[1] in json format. They contain article body texts as well as metadata on articles including a title, a trail text (a short summary that compliments the title), topic tags (multiple per article), the URL, and the word count. We use the article body text as the input to our models and create several targets for our different tasks with the metadata.

For the primary summarization tasks we take the article title and append the trail text as a separate sentence. This forms a relatively high quality 2 or 3 sentence summary of the article based on a qualitative assessment (see figure 3.2 for an example).

For our secondary sequence generation tasks we use the topic tags, this is a variable length sequence of topics that journalists have linked to the document, such as *"Donald Trump"*, *"Brexit"*, *"Theresa May"*, *"addiction"*, *"E3"*, and so on. In the corpus there are 11250 unique tags and 1,205,870 tag events with an average of 4.73 tags per article, figure 3.4 shows the distribution of tags. Figure 3.3 shows an example topic tag sequence, comparing this to the matching reference summary in figure 3.2 we can see that the topic tags align with several keywords in the summary.

For the TL side task, designed to measure information about the global article semantics, we use the section labels (e.g. "politics", "football", "fashion"; 11 in total, see table 3.1) as a multi-class classification target, trained over the latent space article representations. All data were split into training (80%), validation (10%) and test (10%) datasets after filtering (see table 3.1).

[1]Available at https://open-platform.theguardian.com/

”*barclays was lambasted about big bonuses , tax avoidance and speculating on food prices at its annual shareholder meeting yesterday the first since last years libor rigging scandal as the embattled banks new chairman admitted bankers pay was excessive . the appointment of sir david walker as chairman failed to prevent a string of shareholders berating the board about pay . one investor , joan woolard , told the banks directors that anyone who needed more than 0m to live on was just a greedy bastard . barclays , run by antony jenkins since bob diamond quit as chief executive in july in the wake of the 000m libor fine , admitted it paid 000 employees at least 0m last year . woolard called on the board to follow her example and donate their homes to charity . describing herself as a 00-year old widow from lincolnshire , she twisted the slogan adopted by jenkins , who began his presentation to the 000 shareholders assembled in royal festival hall in london with his vision of turning barclays into the go to bank* ....”[truncated]

Figure 3.1: An example article body text.

”*barclays boss promises investors he will crack down on excessive pay . sir david walker berated by shareholders at london agm as beleaguered board tries to shake off libor rate rigging scandal .*”

Figure 3.2: An example reference summary.

”*barclays, banking, antony_jenkins, libor*”

Figure 3.3: An example topic tag sequence.

3.2 Article Texts

For the article body texts, HTML was removed using beautiful soup and text strings were tokenized into words and punctuation symbols. Token count distributions were analysed and the majority of documents contained between 300 and 1200 tokens, with a median token length of 734 (mean 986.96), see figure 3.5. However, there were a few articles with a large token count up to 40,122 and over 16,000 documents had a token count of zero. Further investigation revealed that the documents with a zero token count usually contained videos or images only, indicated by the keywords *video* and *gallery* in the URL. We removed all documents from our dataset that were either video or image galleries or had a token count below 100. We assume that 100 words is the minimum length for a small paragraph carrying enough meaning for summarization to

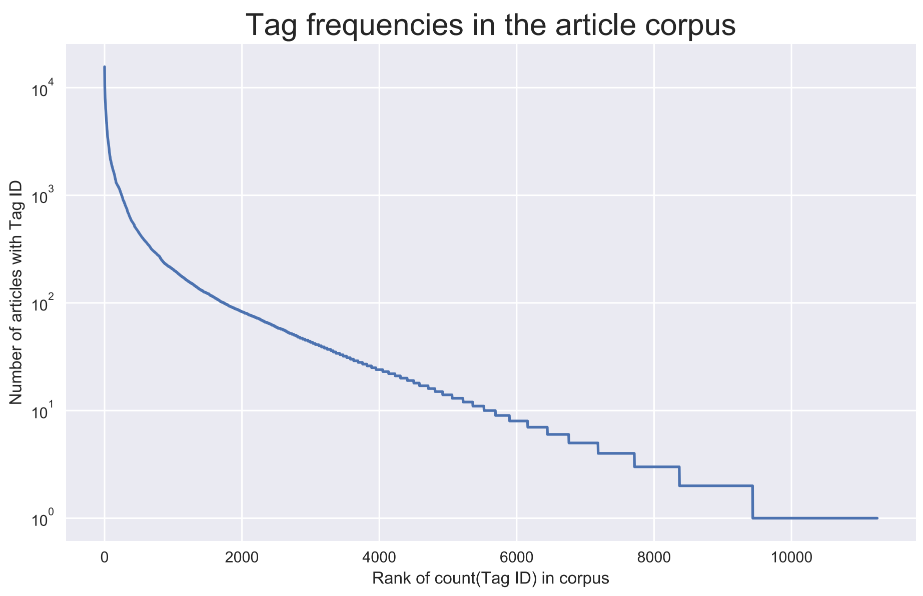

Figure 3.4: Tag frequencies across for all articles. This shows an extreme distribution with a very long tail; it is clear that tags do not follow Zipf's law.

be appropriate. For the purpose of this study we truncate all articles to a maximum of 400 tokens (i.e. word, number, or punctuation symbol), this is to ensure that we do not exceed memory constraints on the GPU and to ensure that training times are reasonable.

After filtering we are left with a dataset of 255,079 documents, having dropped around 20,000 documents, with a total of 251,751,864 symbols (prior to truncating articles). The articles come from different sections on the news site and there is significant class imbalance as seen in figure 3.6. Table 3.1 comprises the exact number of documents per set for each class.

A number of preprocessing steps were taken to prepare the article texts for the tasks and models, and to optimize the efficacy of word embeddings. All body text was converted to lower case to avoid multiple word embedding IDs and non-ascii characters were removed. Punctuation was retained as the embeddings we use do have punctuation embeddings and these may be useful in summarization (for example, information within commas, like this, may be auxiliary and more appropriately represented if punctuation is left intact). Numerals were normalised, replacing digits with a zero. For example, 12.05.2016 became 00.00.0000 and *46 million* became *00 million*. This preserves some semantic information regarding numbers whilst reducing the sparsity of

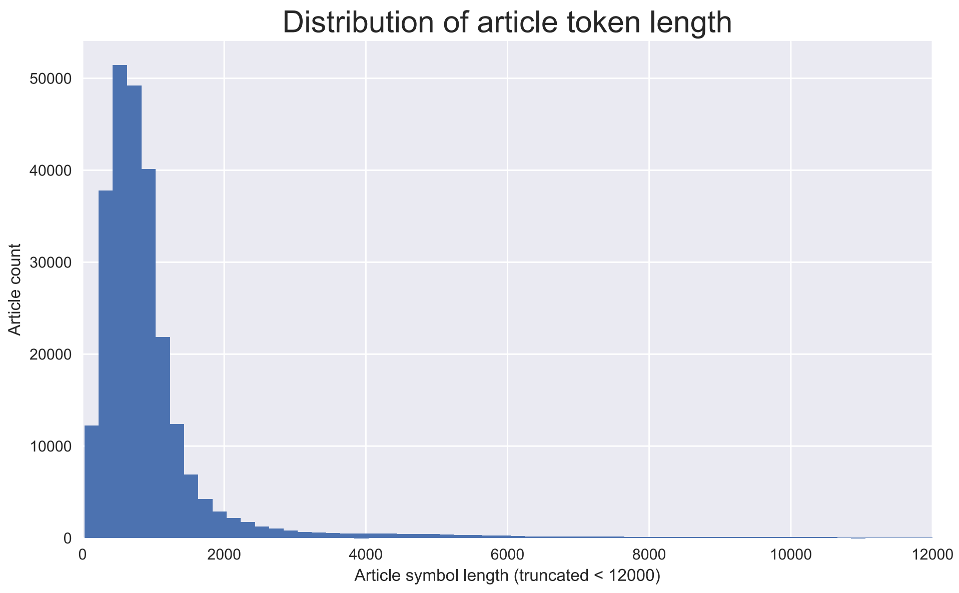

Figure 3.5: Distribution of article symbol length (truncated to 12000).

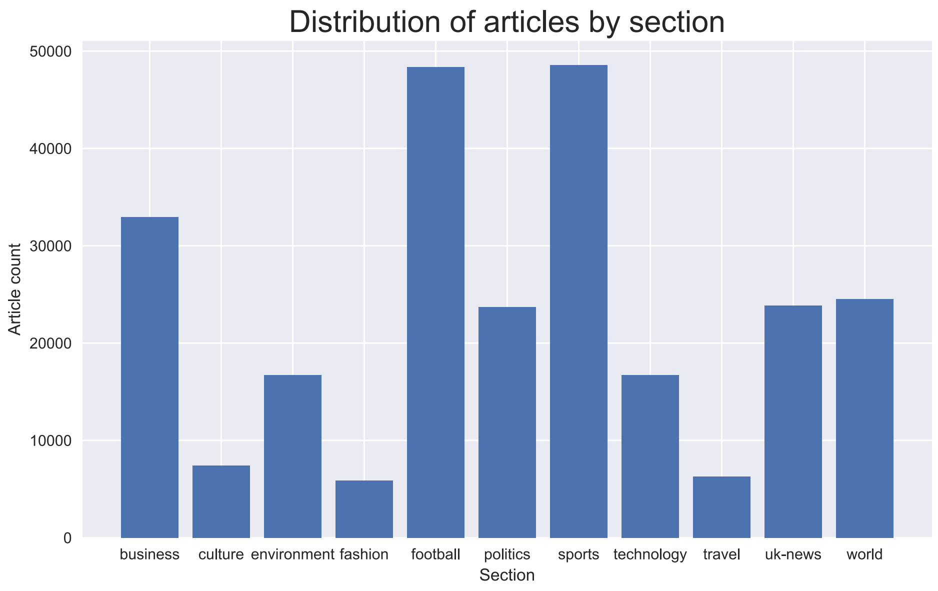

Figure 3.6: Class distribution of preprocessed data.

CLASS	TRAIN	VALID	TEST	TOTAL
BUSINESS	26,356	3,294	3,295	32,945
CULTURE	5,944	742	744	7,430
ENVIRONMENT	13,381	1,673	1,671	16,725
FASHION	4,720	590	590	5,900
FOOTBALL	38,703	4,838	4,838	48,379
POLITICS	18,959	2,370	2,370	23,699
SPORTS	38,879	4,860	4,860	48,599
TECHNOLOGY	13,392	1,674	1,675	16,741
TRAVEL	5,016	627	628	6,271
UK-NEWS	19,089	2,386	2,387	23,862
WORLD	19,664	2,453	2,453	24,528
TOTAL	**204,061**	**25,507**	**25,511**	**255,079**

Table 3.1: Number of documents per class for training, validation and test set.

numbers (this can lead to low probabilities in the language model, since exact numbers may only appear a small number of times).

Chapter 4

Methods

4.1 Overview

Since we introduce a new dataset for summarization, we first aim to establish high quality benchmarks using recent SOTA models to understand what good performance looks like. For the benchmarking work we use a modified fork of the Open-NMT (PyTorch version) library[1] (Klein et al., 2017). We explore two models to establish benchmarks including an ”Vanilla” Encoder-Decoder with global Attention (VEDA), and a Pointer-Generator Network (PGN) with coverage attention. These are discussed in more detail below.

In order to have greater control over the modelling pipeline we also wrote our own implementation for end-to-end sequence-to-sequence models in PyTorch (Paszke et al., 2017). This included scripts for importing and structuring data and vocabularies, importing and loading word embeddings, setting up data iterators and batching, replicating the Encoder-Decoder with attention model[2], methods for evaluating the different tasks, methods for plotting attention visualisations and methods for computing ROUGE metric (using Tardy (2017)). We extended this model to work for predicting tag sequences and for MTL to update the model using a compound loss function. We also implement a model that is capable of taking the latent representations from the encoder and train on top of this a classifier for predicting global document semantics as a way of probing the quality of latent representations. Finally, we train a further model to predict the global semantic classes from an encoder trained on this task specifically, to

[1]The original library can be found at `https://github.com/OpenNMT/OpenNMT-py` whilst our fork is publicly available at `https://github.com/jtizzle36/OpenNMT-py`.

[2]Code available at `https://github.com/jtizzle36/diss_remote_code_seq2seq`. Inspiration was taken from a tutorial by Lo (2018)

establish a ceiling value for the task to compare our model to. We also write methods for transforming, visualising and analysing latent representations using Uniform Manifold Approximation and Projection (UMAP, McInnes and Healy (2018)) a dimension reduction technique.

All code was written in Python and all modelling work was done using the PyTorch library (Paszke et al., 2017) and computed using a single Nvidia GeForce GTX Titan X.

In the following sections we give more details on the methods and work done for different components of our research.

4.2 High quality benchmarks

Since we are introducing a novel dataset for text summarization it is necessary to create high quality benchmarks to better understand the task and what good performance looks like. We create a simple extractive baseline using the LEAD method, where the leading sentences of a document are assumed to form a good summary. Since the majority of our target summarizations are two sentences long we select the first two sentences from each document as the LEAD baseline.

4.2.1 Evaluation

Evaluation methods for natural language generation are still somewhat limited and automated methods still fall short of high quality, comprehensive evaluation for fluent and coherent language and differ significantly from human evaluations (Novikova et al., 2017). In an ideal world we would have used human evaluators in addition to automated metrics, however this was out of scope for the present research due to resource constraints. As such we rely primarily on the automated metric known as ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation (Tardy, 2017). It is somewhat related to the BLEU metric (Papineni et al., 2002) used in machine translation except it is recall-oriented whilst BLEU is precision-oriented. That is, BLEU is concerned with measuring how much the words (or n-grams) in the generated summaries appeared in the reference, whilst ROUGE is concerned with the inverse, how much the words in the reference appeared in the generated summary. As such, BLEU will penalise words or strings in the generation not found in the reference, whilst ROUGE is less concerned with these novelties but more concerned with ensuring the words from the reference appear in the generation at some point. Since we expect a natural language generation system to improvise (indeed, we may hope to see this behaviour) ROUGE is typically considered a more appropriate metric for abstractive summarization. We report both ROUGE and BLEU for unigrams, bigrams and the longest n-gram matching sequence. ROUGE-N is described in general as

$$R-N = \frac{\sum\limits_{S\in(RefSumms)} \sum\limits_{gram_n \in S} Count_{match}(gram_n)}{\sum\limits_{S\in(RefSumms)} \sum\limits_{gram_n \in S} Count(gram_n)} \tag{4.1}$$

We also calculate a metric for novel bigrams. Since the objective of abstractive summarization is to generate summaries without just copying source information as in extractive approaches we also measure the number of n-grams seen in the generated summary that do not appear in the source text. This computes bigrams and trigrams and calculates the proportion of novel bigrams not found in the source text to total bigrams in the generated sequence.

4.3 Language Class & Vocabulary Building

In order to train our sequence-to-sequence models appropriately we create classes for the different tasks that treat our data appropriately. Source texts are truncated to 400 tokens in length (tokens include words, punctuation and numerals), reference texts (i.e. the target summaries) are truncated to 100 tokens. All sequences are bookended with a beginning of sequence token ('$\langle BOS \rangle$') and and end of sequence token ('$\langle EOS \rangle$'). A vocabulary is created by selecting the 50,000 most prevalent tokens in the source and target sequences, which gives a total vocabulary size of 50,004 including the positional tokens we added. We use a shared vocabulary for the encoder and decoder. Tokens not found in the vocabulary are replaced with an unknown token ('$\langle UNK \rangle$'). Sequences that are less than 400 tokens are padded with a pad token ('$\langle PAD \rangle$').

4.4 Word Embeddings

Each token from the input sequences is associated to a vector $w \in \mathbb{R}^d$ via a lookup table. We used GloVe pre-trained word embeddings with a size of 300 dimensions trained on a 6B token corpus comprised of Wikipedia 2014 and Gigaword 5 datasets (Pennington et al., 2014). Of our vocabulary of 50,004 tokens, 4505 words do not have embeddings in the GloVe model. These are converted to unknown tokens resulting in a final vocabulary size of 45,499 words.

All tokens from an input string are mapped to their respective embedding in the first layer of our model. In theory, we could also pre-translate our input and output token sequences to their embeddings, however by including the embeddings as a layer in the model it allows us to backpropagate errors through the embedding layer, which allows us to also modify the embedding vector values to further minimize the loss function. This is known as fine-tuning the embeddings. We fine tune our embeddings allowing the embeddings to move to updated distributions that better reflect their usage in our specific corpus and has been shown to improve performance (Howard and Ruder, 2018). We tested whether using pre-trained embeddings improved quality of summarization and found it improved validation accuracy during training by around 5%. We did not test the effect of fine-tuning embeddings.

4.5 Encoder-Decoder Model

Our core model is the same as the *general* attention model described in Luong et al. (2013). The sequence transduction task in its general form models the probability $p(y_1,...,y_n|x_1,...,x_m)$, mapping to the output sequence $y_1,...,y_n$ from the given input sequence $x_1,...,x_m$. The model itself involves several components described below and shown in figure 4.1. At a high level, the model takes in a sequence of source embeddings,$x_1^{(Emb.)},...,x_m^{(Emb.)}$, passes these through the encoder RNN for m timesteps, where m is the token length of the input sequence and takes the output states of the last layer of the encoder, $h_1,...,h_m$, and stores these for later. It then passes the sequence of target embeddings, $y_1^{(Emb.)},...,y_n^{(Emb.)}$, through the decoder RNN one at a time for n timesteps, at each step it combines the output of the decoder s_n with the preserved encoder states $h_1,...,h_m$ based on the learned context vector c which allows the decoder to selectively attend to the encoder states uniquely at each timestep n. The output of the attention layer is then passed through a softmax w.r.t. the vocabulary resulting in probabilities for all token embeddings in the vocabulary.

For all trained models (unless otherwise specified) we use a two layer, bidirectional LSTM as the encoder with a hidden size of 512 units, that is, each LSTM is 256 units so a single bidirectional layer is 512 when concatenated. We use a single directional single layer LSTM decoder. We use dropout at the input to RNN layers with a probability of 0.3. We use shared embeddings for the encoder and decoder (explained later

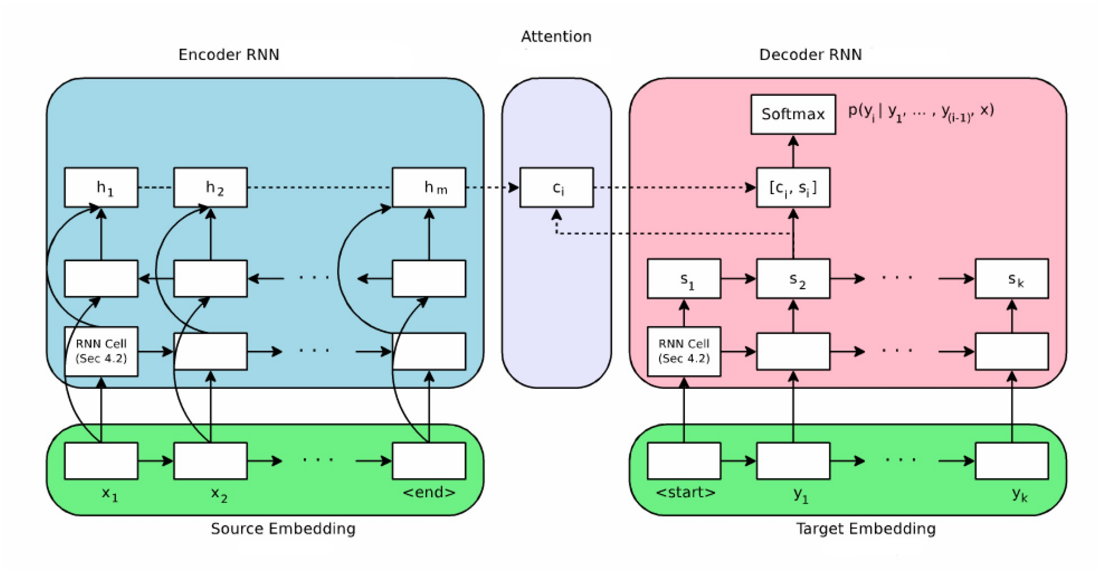

Figure 4.1: Diagram of the Encoder-Decoder with attention model. From Britz et al. (2017).

in this section) and use beam search for our benchmark experiments (also discussed later this section). These hyper-parameters were chosen using a combination of reviewing the literature (particularly Britz et al. (2017)) and by preliminary experiments evaluating performance-resource trade-offs.

4.5.1 Encoder

The encoder takes the form of a stacked 2 layer bidirectional RNN, as described in chapter 4. A bidirectional RNN effectively duplicates the input and reverses it, starting with the last token and ending with the first. This has been shown to dramatically improve performance in machine translation (Sundermeyer et al., 2014). The two representations are then concatenated before being passed to the next layer.

So the encoder function f_{enc} takes the sequence of embeddings $\mathbf{x} = x_1, ..., x_m$ (we refer to these as just x rather than $x^{(Emb.)}$ for readability, from here onwards) and produces a sequence of hidden states $\mathbf{h} = h_1, ..., h_m$, where h_i is the concatenation of the hidden state in both directions $h_i = [\overrightarrow{h_i}; \overleftarrow{h_i}]$.

4.5.2 Decoder

The decoder function f_{dec} is a single directional RNN that predicts the probability of a target sequence $\mathbf{y} = y_1, ..., y_n$ based on $\mathbf{h}$. The probability of each target token

$y_n \in 1,...,V$ is predicted via a softmax over V states based on the recurrent state of the decoder RNN s_n, the tokens already generated, $y_{<n}$, and the context vector c_n, also known as the attention vector. A simple decoder process is as follows

$$s_n = LSTM(h_i, y_{n-1}) \tag{4.2}$$

$$l_n = g(s_n) \tag{4.3}$$

$$p_n = softmax(l_n) \tag{4.4}$$

$$y_n = argmax(p_n) \tag{4.5}$$

If n is the first output in a given sequence and $y_{n-1} = \varnothing$ then we set y_{n-1} to the beginning of sequence token $'\langle BOS\rangle'$. $h_i \in \mathbf{h}$ is some method for selecting from the encoder states such as taking the last state or an average over all states, later we introduce attention for this. g is a function to map the decoder outputs to an vector the same shape as the vocabulary, $g : \mathbb{R}^s \mapsto \mathbb{R}^V$ so that $l_n := g(s_n) \in \mathbb{R}^v$. The $softmax$ function normalizes this to a vector of probabilities over the vocabulary and (in a greedy decoder) we take the *argmax* of the probabilities to find the next most likely token (later we introduce beam search for more long-sighted sequence prediction). Decoding stops when the predicted word is an end of sentence token $'\langle EOS\rangle'$ (or we reach a maximum out put length set by a parameter).

4.5.3 Attention

The attention vector is calculated as a weighted average of the source states $\mathbf{h}$. This modifies equation 4.2 above as follows

$$s_n = LSTM(s_{n-1}, [y_{n-1}, c_n]) \tag{4.6}$$

$$c_n = \sum_j a_{nj} h_j \tag{4.7}$$

$$a_{nj} = \frac{\hat{a}_{nj}}{\sum_j \hat{a}_{nj}} \tag{4.8}$$

$$\hat{a}_{nj} = s_{n-1}^T W h_j \tag{4.9}$$

where c_n is the context vector, a_{nj} are the attention weights, and $\hat{a}_{nj}$ are the raw attention scores. W is a learned weight matrix which transforms the decoder state to a new shape to match the encoder state and serves to optimize the attention mechanism. The form of $\hat{a}_{nj}$ varies in different implementations of attention, we use the *general* form from Luong et al. (2015a).

4.5.4 Shared Embeddings

We also use shared and tied embeddings where appropriate. Tied embeddings are where the input and output embeddings for the decoder are set to be the same, these are used for all models. Shared embeddings are where the encoder and decoder have a common vocabulary and are initialized with the same embeddings. This is used for summarization models. These methods reduce the number of parameters that need to be learned and leads to better training dynamics and improved performance (Inan et al., 2016; Press and Wolf, 2016).

4.6 Training and Optimization

To train the model we calculate the loss for each sequence. The decoder outputs vectors of probability over the vocabulary $p_i \in \mathbb{R}^V$ for each timestep. For a given target sequence $y_1, ..., y_n$ we calculate it's probability as the product of the individual probabilities of each token at each respective timestep

$$\mathbb{P}(y_1, ..., y_n) = \prod_{i=1}^{n} p_i[y_i] \tag{4.10}$$

where $p_i[y_i]$ means we select the y_ith entry from the probability vector p_i from the ith decoding step. During training we know the true sequence so we can maximize the probability of this using the reference sequence by minimizing the cross entropy loss between the target and predicted distributions

$$\begin{aligned} -\log \mathbb{P}(y_1, ..., y_n) &= -\log \prod_{i=1}^{n} p_i[y_i] \\ &= -\sum_{i=1}^{n} \log p_i[y_i] \end{aligned} \tag{4.11}$$

Updating the parameters of the model is done by backpropagation of gradients w.r.t the errors using PyTorch's automatic differentiation module. This updates weights and biases based on our optimization algorithm. We use a dynamic learning rate calculated by Adam. We initialize the learning rate at $1e-3$. We considered other optimization algorithms including stochastic gradient descent and RMSProp but found Adam to be stable and to give favourable training dynamics.

We used a batch size of 16 (due to memory constraints on the GPU) and trained models until convergence. Initially we trained for 190k steps to find a good stopping criterion and identified 140k steps as the ideal point for out benchmark experiments (equivalent to just under 12 epochs). Other experiments were trained for variable steps as specified in the results.

4.6.1 Evaluating the Auxiliary Sequence Task

Since we are predicting a sequence of labels it does not make sense to use ROUGE or sequence generation type metrics. Instead we calculate the precision, recall and F1 statistic as if we were predicting multiple classes.

4.6.2 Beam Search

In out benchmark experiments we use beam search decoding for the Open-NMT trained Pointer-Generator Network models, with number of beams, $k = 5$ (for other experiments we use a greedy decoder). Beam search lifts the greedy assumption that the best sequence is the sequence that maximizes the probability at time n, such that $y_n = argmaxp(y_n|y_<n, x_1, ..., x_m)$, and accepts that there may be better sequences that have a non-maximal probability at time n but have a higher overall sequence probability when considering $p(y_1, ..., y_n|x_1, ..., x_m)$.

In order to accommodate these alternative non-greedy hypotheses we keep track of k candidate sequences (k is the beam size). At each new timestep we have $5V$ possible hypotheses, of which we keep the best 5 based on their probability, and so on. Once every hypothesis terminates we return the sequence with the highest overall probability.

4.7 Multitask Learning

Our MTL experiments test whether training the summarization model with MTL on a semantically related sequence prediction task leads to (a) an improvement in the summaries as measured by traditional metrics and (b) greater semantic information in the latent representations as measured by the semantic class reconstruction side task.

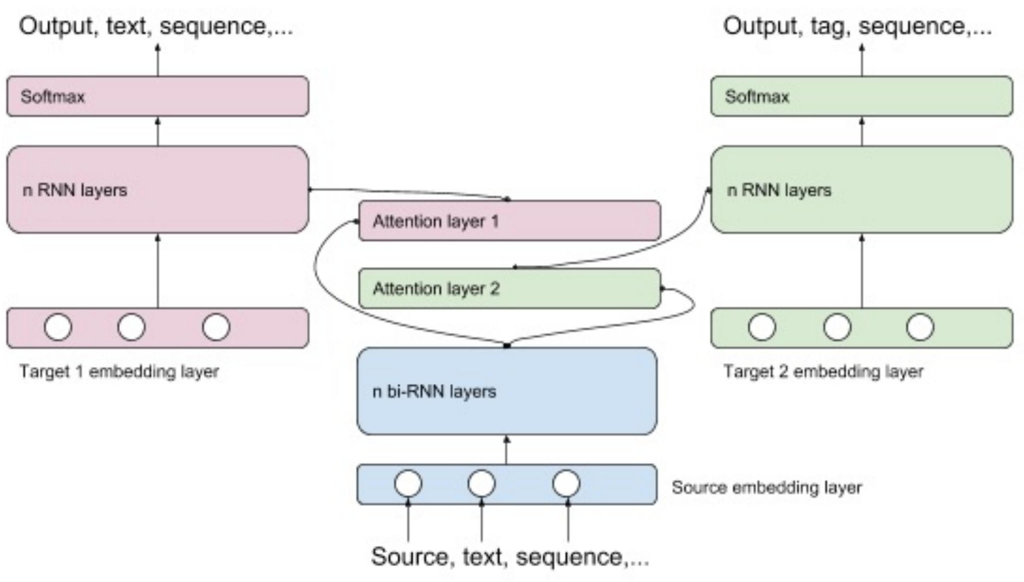

Figure 4.2: Diagram of the multitask Encoder-dual-Decoder with attention model. Internal workings are the same is in figure 4.1.

The auxiliary semantic sequence prediction task is to predict the topic tags associated with each document. These topic tags reflect lower level semantic concepts when compared to the article classes, for instance entities such as people or places. We train a single task model on the topic sequence prediction task using the same model and hyper parameters as for the summarization task to give a single task baseline.

To train the MTL model we amend the architecture above to have a second decoder which also takes the same encoder states as inputs. This decoder is identical to the one described and has it's own attention layer. We optimize the model using a compound loss function. This is done by simply calculating the loss for both tasks, $loss_{summ}, loss_{tags}$, summing these together and backpropagating the compound loss. We also implement a weighted compound loss function using a weight parameter α

such that the new loss is defined by

$$loss = \alpha \cdot loss_{summ} + (1 - \alpha) loss_{tags} \tag{4.12}$$

4.8 Measuring Global Semantics in Latent Space

We create a side task using TL with the trained encoder representations of our sequence transduction models to classify source article semantic class. This task is designed to probe the latent representations learned by the sequence transduction models. Each article is from a *global semantic class*, that is, the document relates to an abstract subject such as ”politics”, ”sport” or ”culture” (11 classes in total, see chapter 3). We test whether the semantic class is recoverable from the learned representations as a measure of semantic information retention from the source text. If one model is better able to recover these classes then we can infer that the representations are of higher quality with respect to capturing abstract semantic regularities from the source documents.

In order to establish an upper bound for this task we must first train a model specifically to perform this task. To this end we train a new model with a similar encoder to our summarization model. This model then passes the last state of the encoder representations through to a single hidden layer with the same number of outputs as there are semantic classes. We pass this through a softmax to get probabilities for each class given the encoder states and we train this model using the cross entropy loss between the predicted and true class for each document.

We then create a model for our task, which is equivalent to the model just described except it receives fixed-weight outputs from the fully trained summarization encoder. In other words, we only update the weights for the class prediction layer when training this model. This allows us to estimate how good a model is at recovering the document semantic classes from their latent representations learned by the encoder, relative to a task-specific encoder (which should achieve an optimal performance given the input information). In this way we can compare different summarization models based on the quality of their latent representations in addition to traditional metrics such as ROUGE. We train this model using a cross entropy loss function over the class prediction.

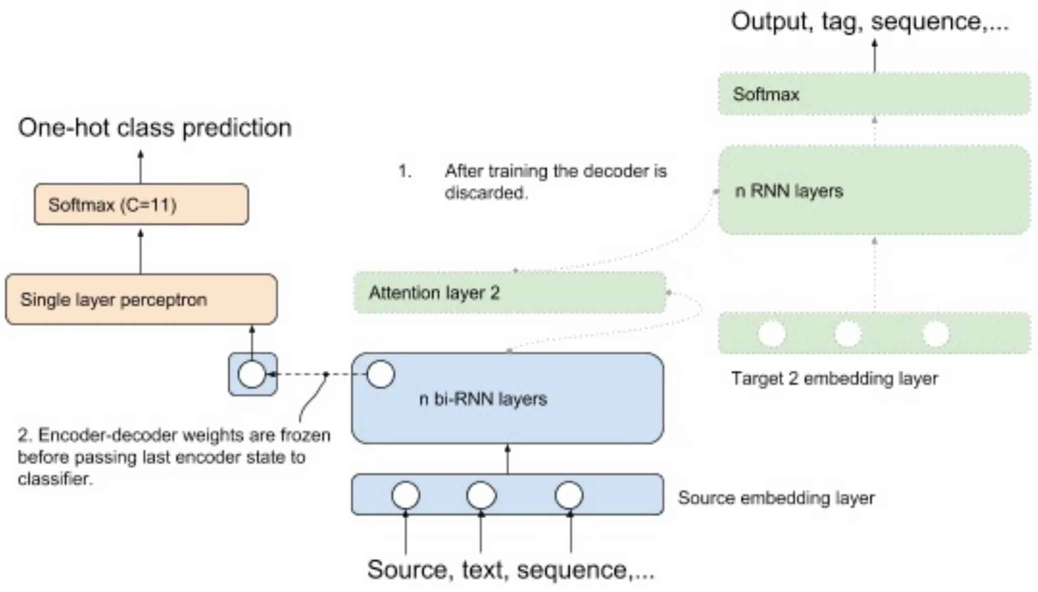
Output, tag, sequence,...
Softmax
n RNN layers
1. After training the decoder is discarded.
One-hot class prediction
Softmax (C=11)
Single layer perceptron
Attention layer 2
Target 2 embedding layer
n bi-RNN layers
2. Encoder-decoder weights are frozen before passing last encoder state to classifier.
Source embedding layer
Source, text, sequence,...

Figure 4.3: Diagram of the semantic classification side task.

Chapter 5

Results & Discussion

5.1 Abstractive Summarization Task Experiments

5.1.1 High Quality Benchmarks

Since our work makes use of a novel dataset it is important to establish high quality benchmarks. In order to do this we trained several recently SOTA models using Open-NMT (pytorch version) (Klein et al., 2017). We create a simple, rule-based, extractive baseline by taking the first two sentences[1] from each article body, referred to as the LEAD summary, see figure 5.1 for an example. Figures 5.2 and 5.3 show an example of a (matching) article body text, which is the input to our model, and a reference summary, which is the target output for our summarization models, respectively.

”*the killing of about 00 people in southern yemen has highlighted an ongoing us backed campaign against al qaidas most active frontline . reports from sanaa and washington confirmed the deaths in air raids and drone strikes against targets in shabwa and abyan provinces , a stronghold of al qaida in the arabian peninsula .*”

Figure 5.1: An example LEAD summary.

We train several baseline models using the Open-NMT pytorch library (Klein et al., 2017) all to 140k training steps (with a batch size of 16): a ”Vanilla” Encoder-Decoder with general attention model (VEDA-onmt, as described in detail in chaper 4, a Pointer-Generator Network with coverage attention, PGN, without pre-trained word embeddings, and a PGN with pre-trained GloVe embeddings. We also train our own im-

[1]using the sentence tokenizer from the NLTK package.

"*the killing of about 00 people in southern yemen has highlighted an ongoing us backed campaign against al qaidas most active frontline . reports from sanaa and washington confirmed the deaths in air raids and drone strikes against targets in shabwa and abyan provinces , a stronghold of al qaida in the arabian peninsula . apparent retaliation followed swiftly , with four yemeni security officers gunned down in the last 00 hours . the aerial attacks described by one experienced observer as massive and unprecedented started on saturday and ended late on monday . the yemeni government said on tuesday night it was investigating whether one of the dead was ibrahim al asiri , a master bombmaker believed to have been involved in several high profile terrorist plots against western targets , cnn reported from sanaa . the death total of 00 was announced by yemens interior ministry ....*"[truncated]

Figure 5.2: An example article body text.

"*yemen conflict highlighted after 00 killed in air raids and drone strikes . campaign against al qaida in the spotlight , as yemeni govt investigates whether prominent bombmaker is among the dead .*"

Figure 5.3: An example reference summary.

"*yemen, middle_east_and_north_africa, al_qaida, drones*"

Figure 5.4: An example topic tag sequence.

plementation of the VEDA-summ model to 85k steps (training truncated due to long training time and resource limitations).

For reporting result we use an identical subset of 3000 randomly selected examples from our test set (a subset is used since the generation process is time consuming and we consider 3000 to be a sufficient sample for statistical analysis). We measure the performance of these models using the ROUGE metrics, R-1, R-2, and R-L, which reflect, as described in chapter 4, recall of unigram, bigram and longest matching n-gram. We report BLEU metrics, B-1, B-2, and B-L, which reflect the precision of unigrams, bigrams and longest matching n-gram. We report the F1 score, the harmonic average, of ROUGE and BLEU. We also report the mean number of novel bigrams found in the generated text string relative to the source text. This will be low if the model is learning to copy sequences from the source text, however a high score is not necessarily good, since it may reflect gibberish that is novel but also nonsensical. In the context of a high rouge score, a relatively high novel n-gram score might imply that the generated text is also innovative.

model	F1-1	B-1	R-1	F1-2	B-2	R-2	F1-L	B-L	R-L	Novel bigrams (%)
REF	-	-	-	-	-	-	-	-	-	63.62
LEAD	30.69	25.79	**41.36**	11.07	8.90	**16.25**	23.32	22.17	**35.62**	0.0
VEDA-onmt	28.09	32.17	25.91	6.95	7.45	6.76	24.41	29.33	23.63	**74.71**
PGN wo emb.	33.93	33.45	35.71	11.44	11.16	12.19	29.50	29.98	32.07	26.29
PGN	**34.34**	34.02	35.94	**11.82**	**11.60**	12.51	**29.87**	30.48	32.28	29.20
VEDA-summ	29.75	**36.72**	26.60	8.37	9.62	7.86	25.00	**33.42**	24.14	49.12

Table 5.1: Results for baseline models measured by ROUGE (R-1, R-2, R-L), BLEU, (B-1, B-2, B-L), F1 statistic (harmonic mean of ROUGE and BLEU), and novel bigrams (%). 1, 2 and L refer to unigram, bigram and longest n-gram, respectively. We include the reference (REF) to show novel bigrams, LEAD extractive baseline, Vanilla attention model, VEDA-onmt, Pointer-Generator networks (PGN; with and without embeddings) and our implementation of the vanilla attention model, VEDA-summ. Best scores are in bold.

Table 5.1 shows results of the benchmark models. We can see that the reference summaries are clearly not extractive with 63.6% of bigrams being novel and therefore not observed in the source text. The simple LEAD baseline of extracting the first two sentences performs quite well and has best overall performance as measured by the ROUGE metrics, however these summaries are long (mean avg. 62.16 tokens), almost twice the average length of the reference summaries (mean avg. 35.58 tokens), and have zero novel bigrams since they are (by design) entirely extractive. The F1 score can be considered a good summary statistic of the two if we consider these aspects equally important. Although both of these metrics are important for abstractive summarization, typically ROUGE is considered the most important of the two. The best performing model according to the F1 statistic is the PGN with pre-trained GloVe word embeddings. Although worse than the LEAD baseline by ROUGE metrics, it is considerably better on BLEU, meaning that it is better at excluding irrelevant tokens but less good at including all tokens from the reference. It is clear that including pre-trained word embeddings is beneficial to performance, in the PG model it gives modest gains of $1-3\%$ across all metrics. The VEDA model trained with the Open-NMT library performs adequately overall, although it scores noticeably worse on recall metrics when compared LEAD and our best performing model. It does score highest on the novel bigrams metric, however this may be due to the generation of non-adequate or non-fluent strings.

Our hypothesis was that high quality benchmarks could be trained for this novel dataset. The main dataset in use as a benchmark for abstractive summarization is the CNN / Daily Mail as presented by Nallapati et al. (2016). Recent SOTA scores on this dataset as measured by the ROUGE-1 metric range from 35.46 to 41.20 with the Pointer-Generator with coverage model scoring 39.53. On the novel Guardian news dataset presented here, PGN scores 35.94 without significant hyper-parameter search and tuning. We conclude that this dataset appears to be a good candidate for abstractive summarization, furthermore we believe that this dataset presents an interesting challenge due to the high proportion of novel bigrams contained in the reference summaries. Perhaps a weakness of the dataset is that the LEAD baseline scores the highest ROUGE scores, suggesting that models that can learn to extract, either explicitly as in extractive methods or implicitly as a learned strategy, from the first two sentences alone will do well on this task. Perhaps it is interesting that the PGN does not learn this strategy. We suggest this may be due to the loss function providing a different objective to the ROUGE score. We discuss the contribution of the loss function in more detail later in this section.

5.1.2 Our model

Our implementation of the VEDA model, VEDA-summ, outperforms the Open-NMT model, VEDA-onmt, across all ROUGE metrics, has best in class scores for BLEU and shows moderately high novel bigrams, significantly more than the PGNs but fewer that the VEDA-onmt model and the reference. Overall we take this as a good indication that our implementation is sufficient, and whilst not as good as SOTA systems, it forms a good baseline for our further experiments with alternative tasks and multitask learning (MTL).

Figure 5.5 shows a plot of attention weights at each decoder step over the source text (cropped due to size constraints, full figure available in the digital appendix). The attention mechanism can be seen functioning as a monotonic copy function across the bright diagonal of the figure. Other parts of the figure show word substitutions, such as $britain \rightarrow uk$ at the second output.

Model	Text
REF	”london night tube : meet the people who travelled on the first night . a few hours on the victoria line s new night service yield music , laughter , selfies , mice and an encounter with london mayor sadiq khan .”
LEAD	”it was the night that london s underground finally became a 00-hour service , or , as one driver put it in an announcement : it s 0am and you lot are still on the tube . the victoria and central lines are the first two underground lines launching a night service .”
VEDA-onmt	”london night : london s night bus . the london night is a night of the night , but the london s night of the london night , the london s night , the london s night tube is a f**king departure .”
PGN wo emb.	”london 0000 : victoria and central lines are first two underground lines . london s underground finally became a 00-hour service , or , as one driver put it in an announcement : it s 0am and you lot are still on the tube .”
PGN	”london night tube : what the experts say . the victoria and central lines are expected to follow suit before the year is out . it s a momentous occasion for all of us londoners .”
VEDA-summ	”london tube strike : what we learned this week . from the tube to london to london 0000 , here are five things we learned from the weekend s tube network .”

Table 5.2: Reference and generated summaries for a random test example (profanity filtered). Source text in the Appendix (table A.1).

5.1.3 Qualitative Analysis of Generated Summaries

Table 5.2 shows the reference summary and generated summaries for our baseline models given a randomly selected source text. This example highlights the complexity of the task, the reference includes a complex construction involving a topic, the London night tube, followed by a colon and a subtopic, meeting people traveling on the first night. The second sentence references various concepts that are discussed

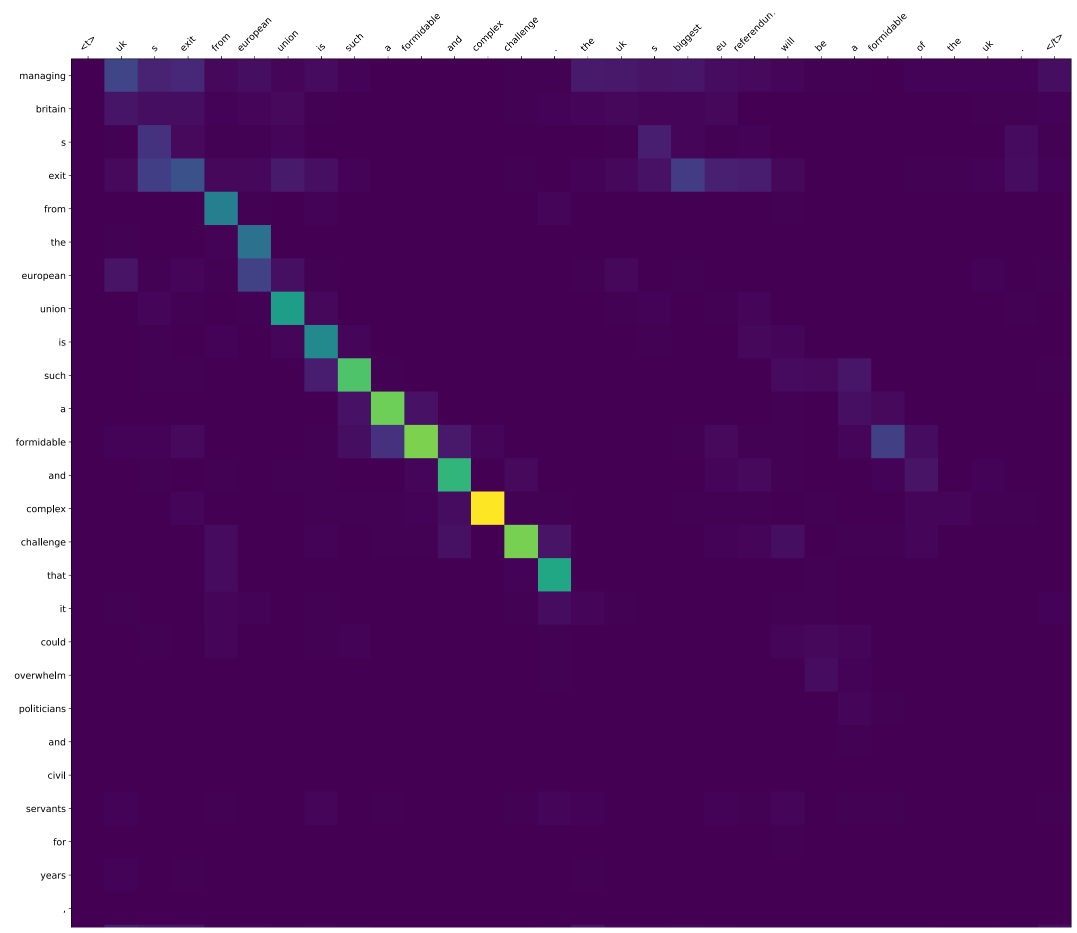

Figure 5.5: Attention weights for each decoder step over source text (crop from full figure). Generated output is on the top, source text on the left.

throughout the whole article. We can see that the LEAD summary does refer to the main subject of the night tube (or at least the 24 hour service) and it identifies the time frame scope, that it was the first night, however it does not do a good job of capturing the context from the second summary or the subtopic of meeting the riders. It contains quite a lot of redundant information and highly specific details not normally found in a summary. We can see from the model-generated summaries that, despite comparable ROUGE scores to the LEAD summary, some of them are very poor summaries in terms of adequacy and most contain disfluencies in parts, or across the whole summary. In particular, the VEDA-onmt model is particularly poor; it captures the 'topic-colon' structure but suffers from repetition of the main subject and refers to a night bus rather than the tube, presumably due to the language model favouring this sequence, since "night tube" is a less frequent bigram in the training data. The next sentence is not fluent due to significant confusion of appropriate subject-object relations, such as "the london night is a night of the night"; arguably the syntax is valid but the sentence is essentially nonsense. Interestingly it copies a profanity from the source text, which is likely to be a rare work in this corpus, and almost certainly does not appear in any reference summaries. It is likely to be a result of the mechanism that copies the work with highest attention when the generative model produces an unknown token. It remains unclear as to what factors lead to attention being focused on one word over another. The PGNs are a significant improvement over the VEDA-onmt model and are, for the most part, fluent summaries. The PGN with pre-trained embeddings, in particular, accurately captures the 'topic-colon-subtopic' structure, although the subtopic is only partially accurate; it identifies that it is a groups' opinions, however it incorrectly refers to the group as experts. This may be an overgeneralisation due to the corpus including many expert reports. The second sentence is fluent but factually incorrect as it substitutes the line names, mixing up the currently active night lines for future planned night lines. In the last sentence it actually does a good job of capturing the gist of the reference without the details. The PGN model without embeddings does a reasonable job although it inaccurately identifies the main topic as "london 0000", which is probably an over-generalization due to the large number or articles about the London Olympics. Both the location and time period are important in the article however it misses the main subject of the night tube, leading to the false implication of an entirely new underground. In the next sentence it does correctly refer to the 00-hour service (not 24, due to preprocessing) but ends with a flippant quote, unsuitable for a good summary. The summary from our model identifies the main topic as the "london tube strike", which is incorrect and likely due to over-fitting / under-regularization, and the subtopic of "what we learned this week", which is somewhat appropriate. The second sentence is fluent and makes references to the tube and new things we have learned but is not an adequate summary of the article.

Table 5.3 shows summaries generated for the source text our vanilla model performed best on according the the ROUGE-1 and ROUGE-2 F1 statistics. Inspecting this example it is clearly quite easy, with the second sentence of the reference being entirely extractive (as seen in the LEAD summary first sentence) whilst the first sentence can be formed entirely with just deletion operations on the second sentence. It is surprising the PGNs do not perform better on this example, the PGN with pre-trained word embeddings in particular performs poorly. Our assumption was that models would tend to do well or poorly on the same examples so we are surprised by this example. In order to investigate this further we analyse correlations of ROUGE scores across the different models.

Model	Text
REF	”crystal palace set to break club record fee to sign james mcarthur for 0m . crystal palace are set to break their club record fee to sign the wigan midfielder james mcarthur for 0m .”
LEAD	”crystal palace are set to break their club record fee to sign the wigan midfielder james mcarthur for 0m . mcarthur s transfer would surpass the former club record fee paid to peterborough last summer for dwight gayle , and would cost palace the same figure wigan agreed with leicester city before they opted instead to secure esteban cambiasso on a 00-month deal .”
VEDA-onmt	”crystal palace to sign wigan athletic for 0m fee for 0m . crystal palace are set to make their 00m signing of wigan midfielder wilfried zaha for 0m to sign a 00m fee for 0m .”
PG wo emb.	”crystal palace set to break record fee to sign james mcarthur for 0m . mcarthur s transfer would surpass the former club record fee paid to peterborough last summer for dwight gayle , and would cost palace the same figure wigan agreed with leicester city before they opted instead to secure esteban cambiasso on a 00-month deal .”
PG	”football transfer rumours : crystal palace to break james mcarthur on record fee ? adl ne guedioura to wigan ? — glenn murray to wigan ? — neil warnock to nottingham forest ?”
VEDA-summ	”crystal palace set to break record fee for 0m . crystal palace are set to break their club record fee to sign the wigan midfielder james mcarthur for 0m .”

Table 5.3: Reference and generated summaries for the best summary generated by our model as measured by ROUGE-2 F1 statistic (0.913). Source text in the Appendix.

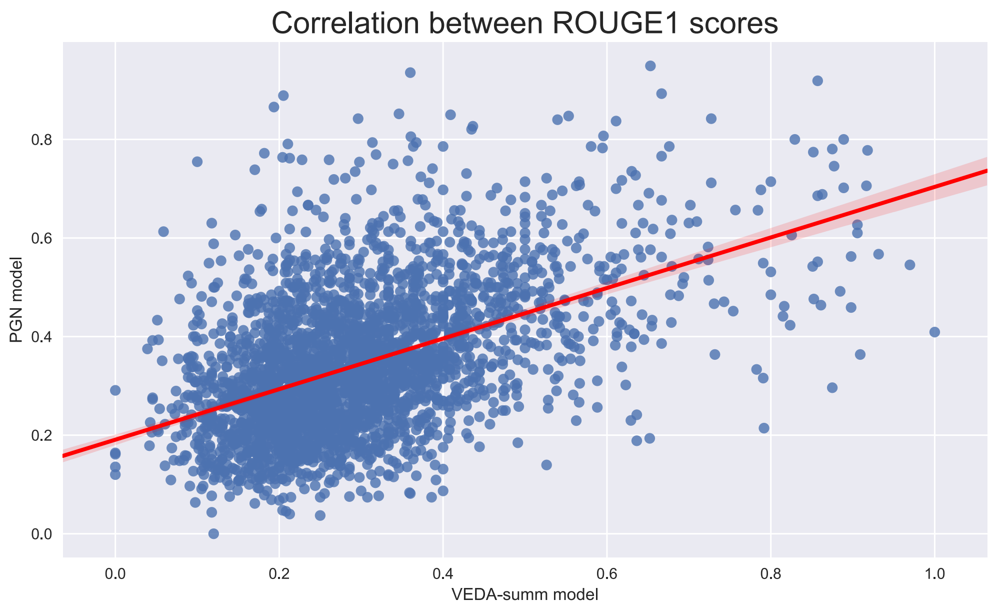

Figure 5.6: ROUGE-1 F1 scores for the PGN (y axis) and VEDA-summ model (x axis). The plot shows a significant moderate correlation (0.51, r=0.49).

5.1.4 Analysis of ROUGE

Figure 5.6 shows a regression model fit to the ROUGE-1 scores for the PGN and the VEDA-summ model, revealing a moderate correlation (0.51, r=0.49) between the two scores. This somewhat confirms our assumptions that, in general, the two models produce similar ROUGE scores for the same source texts, however it is not a strong correlation and it is clear from the distribution that there are a number of outliers that may score very high with one model and very poorly with the other. This suggests that, despite relatively similar architectures, there are significant performance differences due to the novelties across the models. It is not the case that one model simply does a bit better across all examples, it seems that one model might completely fail for a given example whilst the other model can perform well on that same example. Figure 5.7 shows a pair plot with regressions for ROUGE-1 comparisons for all models. We can see that, as we might expect the two PGN models have a particularly strong relationship whilst LEAD has a particularly weak association with all other models; but strongest with the PGN, which makes intuitive sense since these modules also include extractive functions. It is clear from this figure that there is a great deal of variation across the models and there is not strong agreement across most of them. This is useful information, for instance it tells us that in many cases poor performance is not simply down to difficult examples that are hard for all models, but may be down to an incorrect summarization function for a given model.

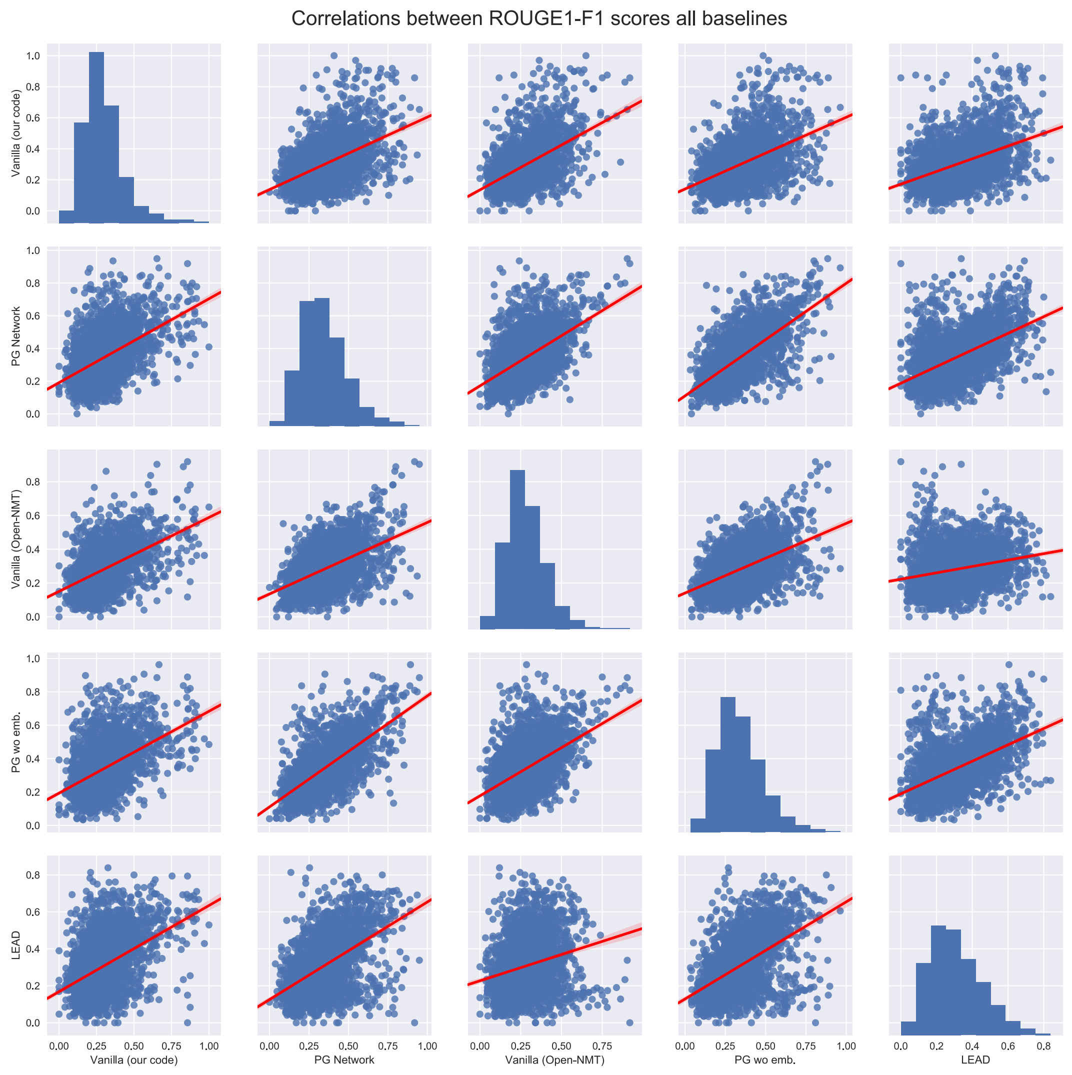

Figure 5.7: ROUGE-1 scores for all model pairs.

5.1.5 ROUGE by section

Whilst inspecting top ROUGE results we noticed that many of the best scoring samples were from the same two sections: football and sports. To investigate this further we calculated ROUGE-1 scores for each section across different models, shown in table 5.4. By computing z-scores we can inspect whether the patterns are consistent across the different models. Firstly we notice that there appears to be a relationship between the trained models, VEDA-summ and PGN, and the class imbalance shown in figure 3.6. We suspect that the model is biased toward accurate summarization of the more prevalent class. This would fit with the mechanism by how we update our weights. This could be addressed using a weighted loss function that penalizes errors based on additional information such as the newspaper section and training sample prevalence.

We also notice that the LEAD method reveals biases that cannot be due to training dynamics. This highest scoring sections for the LEAD method are UK news, World news and business sections which have close to mean performance in the trained models. It is likely this is due to structural regularities in the data; these sections are ”canonical” news and are perhaps more likely to have a simple message that is included in both the summary and the leading sentences of the article. Sections that perform worst with the LEAD method, travel and fashion, also perform worst on the trained models. These sections reflect less cannonical news articles. It was beyond the scope of our research to investigate this further however, inspecting the performance of summarization by categories like this might reveal interesting factors dictating generation quality.

Our hypothesis was that we would be able to train a high quality model using our own code that would perform equivalently to the baseline models trained with the Open-NMT library. Although our VEDA-summ model does not perform as well as the PGN it outperforms the equivalent VEDA-onmt Open-NMT model. We also show that, contra to expectations, models do not always generate low quality summaries on the same source texts. This suggests that poor performance is not driven by exogenous sources of difficulty (i.e. the source is just hard to model) but endogenous sources of variation in how the models generate summaries.

Section	VEDA-summ	z-score	PGN	z-score	LEAD	z-score
business	25.36	0.25	36.79	0.61	45.32	0.83
culture	21.38	-0.64	35.03	0.15	31.06	-0.87
environment	24.00	-0.05	34.14	-0.08	37.47	-0.11
fashion	17.22	-1.56	30.55	-1.01	24.79	-1.62
football	33.20	2.00	38.32	1.00	41.19	0.34
politics	26.99	0.62	36.64	0.57	44.28	0.70
sport	27.28	0.68	35.65	0.31	39.27	0.11
technology	22.58	-0.37	33.87	-0.15	37.02	-0.16
travel	18.00	-1.39	24.68	-2.51	25.15	-1.57
UK news	26.07	0.41	37.68	0.84	49.68	1.35
world news	24.48	0.06	35.51	0.28	46.84	1.01

Table 5.4: ROUGE-1 by newspaper section and z-scores for each model. There are clear differences across sections and these differ by LEAD versus trained models.

5.2 Topic Tag Sequence Generation Experiments

We train the VEDA model, as above, on the topic tag sequence generation task, using our implimentation, VEDA-tag, and Open-NMT, VEDA-onmt-tag, to see if this model architecture is capable of learning other non language-like sequences. We do not replicate the PGN model since this makes no sense for this task. We measure performance using precision, recall and the F1 statistic over predicted sequences. Due to resource constraints we were unable to train our model (which takes significantly longer per step) for the full 140k steps we trained the baseline models for. To account for this we also report results for the Open-NMT model at equivalent training steps. Table 5.5 contains the results; although our model, VEDA-tag, does not achieve the same performance as the fully trained Open-NMT model it does perform significantly better than the Open-NMT model trained to equivalent steps (15k with a batch size of 16). The models perform well in terms of precision, the best score being 75.91, but fall short on the recall metric, only achieving a best score of 51.90. We suspect this is due to the model failing to generate rare tags. As discussed in chapter 3 the tag distribution is extremely long tailed, as such the probabilities for rare tokens are going to be very low. We perform an analysis to check this hypothesis by calculating the average prevalence of tags found in the predicted tags vocabulary versus tags not found in the predicted tags vocabulary. We find that the tags that are predicted have a median prevalence (count of occurrences over total tags) of $2.28e-04$ whilst the median prevalence of tags not found in the predicted vocabulary is $4.37e-06$. We report the median rather than the mean values due to the non-normal distribution of the data, however the mean values show the same relationship but with more extreme values. This same issue is seen with language modelling as well and there are methods to try to resolve it. We could subsample the documents (or over-sample) to artificially alter the frequency statistics. This could lead to improved tag predictions but may also lead to false positives. Alternatively we could use a back-off method for all rare labels during decoding to increase the likelihood that they are generated. We do not explore these improvements further since the objective of this task was not to create particularly accurate topic tag predictions but to develop this task to be used as an auxiliary task during multitask training for text summarization. We have demonstrated that the modelling approach works well enough for our further experiments. We were somewhat surprised that this did work quite well as we expected more issues with repetition and with the fact that the reference tags do not have a strict sequential nature like natural language. In fact the model appears to learn that tags are not repeated.

model	F1	precision	recall
VEDA-onmt-tag 140k	**58.73**	**75.91**	**51.90**
VEDA-onmt-tag 15k	16.94	0.2263	14.80
VEDA-tag 15k	42.49	66.65	34.77

Table 5.5: F1 statistic, precision and recall for predicted tag sequences (we report as a %).

Figure 5.8 shows a crop of the the attention weights at each decoder step (cropped due to size constraints[2]). Interestingly we do not see expected patterns for attention mappings between source words and decoded tags, for instance in the plot we observe strong attention between source word ”overwhelm” and the tag ’eu_referendum_and_brexit’ but very little attention over ”britain s exit from the european union”. This is fairly consistent across examples. Speculatively, this could be due to overfitting due to either our training dataset being too small or poor regularization of the network during training. If the training data are to small then certain words might happen to be highly associated

[2]full scale plots are included in a digital appendix

Model	Text
LEAD summary	”managing britain s exit from the european union is such a formidable and complex challenge that it could overwhelm politicians and civil servants for years , senior academics have warned . theresa may has announced she will trigger article 00 the two year process of negotiating a separation from the eu by the end of march next year .”
REF	”eu_referendum_and_brexit, european_union, foreign_policy, politics, uk_news, academic_experts, business, europe, world_news”
VEDA-onmt-tag 140k	”eu_referendum_and_brexit, european_union, europe, foreign_policy, politics, uk_news, world_news”
VEDA-onmt-tag 15k	”olympic_games_0000, sport, olympic_games”
VEDA-tag 15k	”eu_referendum_and_brexit, uk_news, european_union, foreign_policy, politics”

Table 5.6: LEAD summary (for context) and generated topic tag sequences from a randomly selected test set source text. The 15k VEDA-onmt-tag model was predicting tags about the wrong topic entirely. On further inspection this is consistent across examples and appears to be due to over-generalization at early stages of training.

with a document topic, for instance ”overwhelm” and ’eu_referendum_and_brexit’. In a larger corpus we would observe the word ”overwhelm” in more contexts, which would help regularize this type of overfitting. We do implement dropout (p = 0.3) on the source texts and on the outputs of our RNN layers which should address this single source word overfitting however due to the relatively short training scheme used on these models (3 epochs) it is possible that this dropout is not having a strong enough effect. There are other variants of dropout that have been shown to be highly effective for RNNs, such as Variational dropout (Gal and Ghahramani, 2016) or Zoneout (Krueger et al., 2016), which would likely improve performance here. Implementing these was sadly beyond the scope of the present research.

Our hypothesis for the TTSG task was that the same model architecture used for

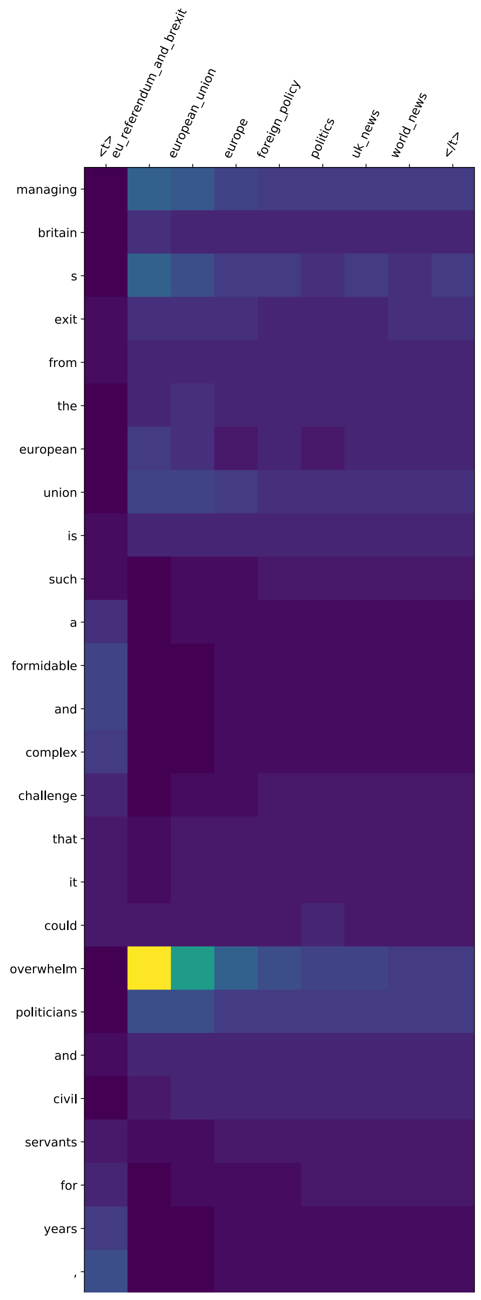

Figure 5.8: Attention weights for each decoder step over source text (crop from full figure). Generated output is on the top, source text on the left.

generating text summaries could also be used to model sequences of topic tags. We have not found reference in the literature to such a task so we consider this to be an example of a novel application of the encoder-decoder model for sequence generation. As such we do not have a benchmark to compare our findings to, however the results suggest that our model is able to perform this task reasonably well within the context of the extreme distribution of the topic tags. Qualitative inspection of the example outputs show that our model is learning appropriate topic tags and should be suitable as an auxiliary task in the MTL experiments we intend.

5.3 Multitask Sequence Generation Experiments

We trained several MTL models due to unsatisfactory outputs (see figure 5.9 for examples and tables 5.7 and 5.8 for results), however none produced fluent, adequate or accurate results for either task. We discuss what we tried to improve these models and candidate causes. Given the results for the two single tasks in isolation we were hopeful that the MTL models may perform well. Our baseline model was similar to the model architecture used for the single tasks but with two decoders run in parallel as described in chapter 4 and figure 4.2. The model outputs were inadequate, producing highly repetitive sequences, often with a single token repeated for several or all decoder timesteps. Initially we thought this might be due to a resource bottleneck in the model and competing objectives between the two tasks. In order to improve the quality of the summaries we implemented a weighted compound loss function as described in chapter 4. We trained a new model with the weight parameter $\alpha = 0.7$, which would give more significance to the loss of the summarization decoder and should improve performance. According to the training metrics there was improvement in training accuracy however when inspecting the outputs and evaluating using ROUGE the outputs are still poor quality. We trained a third model with two further modifications; we removed the bookend sentence tokens $('\langle t\rangle','\langle/t\rangle')$[3] as these were often the subject of repetition. We reasoned that, since they are at the beginning and end of every sentence their probability would be very high and therefore dominating the decoder's probability-based outputs. We also adjusted the weight $\alpha = 0.85$ to further bias the model toward favouring the summarization task. As can be seen from the examples in figure 5.9 neither of these changes lead to a discernible improvement in the adequacy or fluency of outputs of the MTL model. We do see differences in the results based on our changes, results in tables 5.7 and 5.8, summarization quality as measured by ROUGE is higher in the equal loss model. This is a surprising finding since we would expect weighting the model toward the summarization task would improve summaries. It does appear to increase the unigram precision significantly whilst decreasing the recall. We believe that this may be related to the loss function and future work might

[3]these have been shown to improve performance (Klein et al., 2017)

consider using a weighted cross entropy to favour recall over precision, since this is considered more important in the task of summarization (Panchapagesan et al., 2016). Tag sequence prediction is significantly improved by removing the bookend tokens and weighting the loss toward the summarization task. Although the majority of this benefit is down to removing the bookend tokens there does appear to be a modest improvement from weighting the task as evidenced by our model that only includes the weighting parameter $\alpha = 0.7$. This is counterintuitive since we weight the loss towards the summarization task and away from the tag sequence prediction task so, if anything, we might expect performance on this task to go down. This creates a trade-off in our model between performance on the primary summarization task versus performance on the auxiliary tag sequence prediction task but in the opposite direction to our expectations.

In the context of our approach, the addition of an auxiliary task is hindering the learned representations such that it is unable to perform either task as well as in a single task approach. This could be due to several factors which we will now discuss.

model	F1-1	B-1	R-1	F1-2	B-2	R-2	F1-L	B-L	R-L
LEAD	30.69	25.79	41.36	11.07	8.90	16.25	23.32	22.17	35.62
VEDA-summ	29.75	36.72	26.60	8.37	9.62	7.86	25.00	33.42	24.14
MTL equal loss	**12.27**	71.28	**6.84**	**0.13**	0.46	**0.08**	**6.61**	**68.64**	**6.53**
MTL weighted loss $\alpha = 0.7$	6.49	**95.88**	3.36	0.0	0.0	0.0	3.36	68.10	4.29
MTL weighted loss $\alpha = 0.85$	7.58	66.65	4.08	0.11	**0.50**	0.07	3.92	64.51	3.90

Table 5.7: Results for the MTL summarization task.

model	F1	precision	recall
MTL equal loss	0	0	0
MTL weighted loss $\alpha = 0.7$	0.23	0.80	0.14
MTL weighted loss $\alpha = 0.85$	**26.20**	**61.20**	**18.50**

Table 5.8: Results for the MTL TTSG task.

Our primary concern is with the loss function. This could be ineffective for several reasons in the context of sequence transduction, and made worse by resource constraints in the MTL setting. When observing the training dynamics the loss function is clearly doing what it ought to, since the training accuracy increases and the loss decreases. However, the loss function we use, the cross entropy loss between the predicted and target sequences, is only a proxy measure of transduction quality. To illustrate this, imagine we have a summary that repeats an important individual's name several times. If the model fails to form coherent summaries it might instead learn to repeat certain keywords from the source text in order to blindly minimize the cross entropy loss. We suspect this is what is happening in out case. One solution to this would be to use a more accurate proxy of transduction accuracy such as ROUGE for generating a loss function (e.g. 1 - ROUGE-1). The downside of this approach and the reason it is not typical is that the ROUGE computation is very slow compared to calculating the cross entropy and so it is practically unsuitable.

It might also be the case that the compound cross entropy loss function is not effective for learning good representations for these tasks together. We have two concerns here. Firstly the compound loss is backpropagated through the whole model, including both decoders. Although this is typical for multitask training (e.g. in Anastasopoulos and Chiang (2018), we question whether this introduces complications for the decoder training dynamics, since both decoders are also being updated by the compound loss function which includes information about the other task, which is irrelevant for the decoder stage. Ideally, the docoders would update based only on the loss relevant to their own task but the gradients for these would be combined for updating the encoder layer. Practically this is challenging, since we utilize the Pytorch auto-differentiation function to handle updating weights efficiently and this proposal would require a custom differentiation module to handle combined gradients.

One solution would be to try a different multitask model architecture, for instance rather than using hard parameter sharing, as we do in our model, soft parameter sharing like that used by Luan et al. (2017) might lead to better results. Sadly, this was beyond the scope of the present research.

Our hypothesis for the MTL experiments was that the auxiliary task of generating topic tag sequences for the documents would help improve the generated summaries, since it would encourage the model to attend to different tag topics during sequence generation. It seems likely from our results that our main hypothesis for the MTL experiments is false. Although it is possible that the experiment failed due to archi-

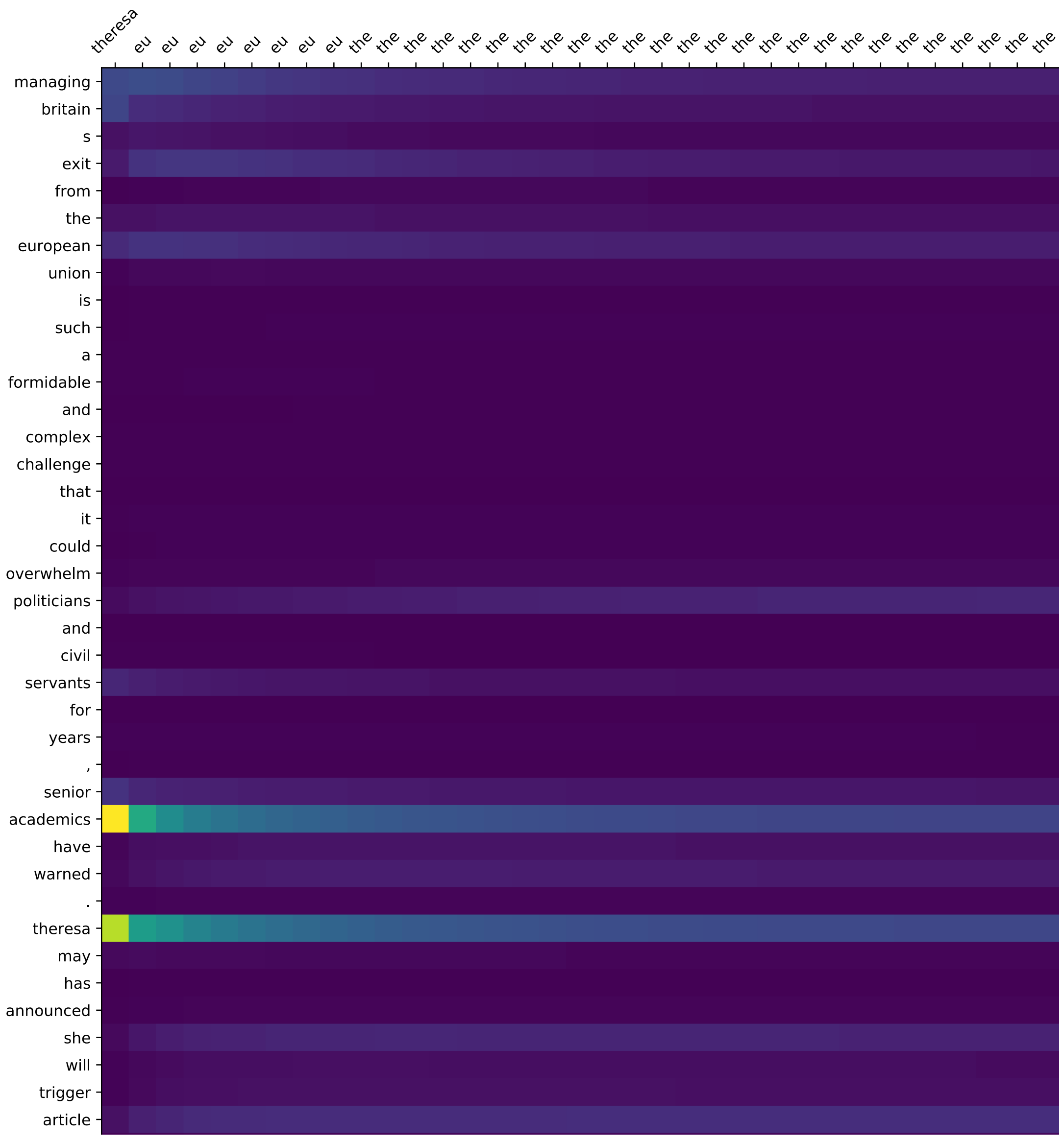

Figure 5.9: Cropped plot of attention weights for the summarization decoder of our MTL model. Generated output is on the top, source text on the left.

tectural or procedural parameters such as the loss function, our results point toward a detrimental effect of MTL abstractive summary generation with this auxiliary TTSG task. Furthermore, this detrimental effect is not moderate but is catastrophic for the model outputs.

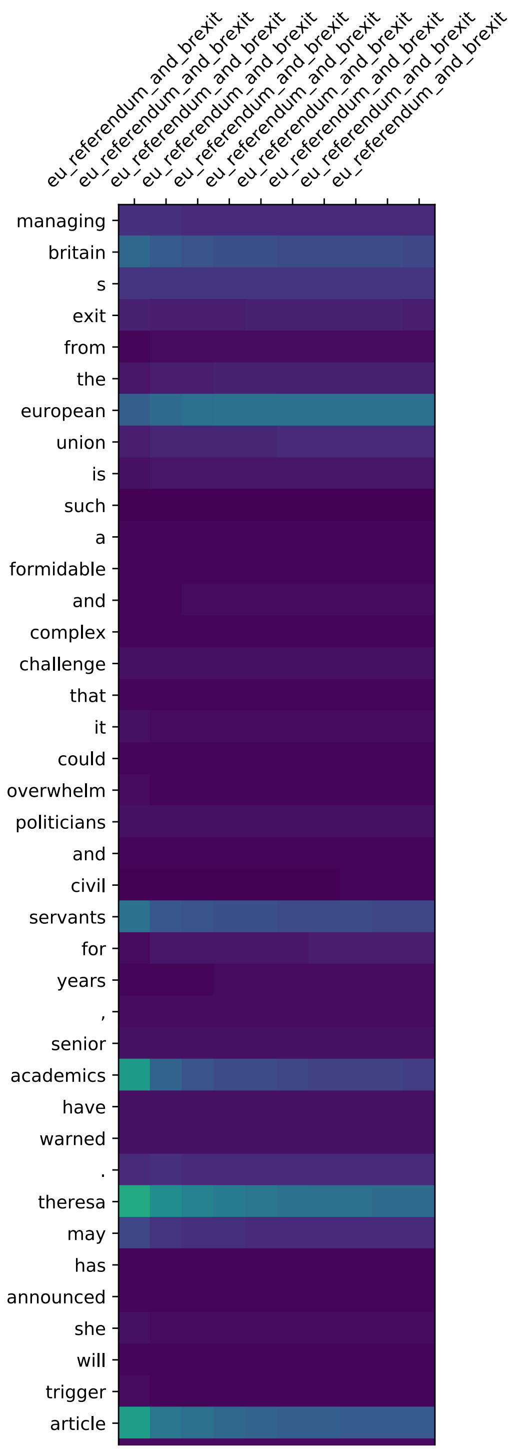

Figure 5.10: Cropped plot of attention weights for the tag sequence decoder of our MTL model. Generated output is on the top, source text on the left.

5.4 Latent Representation Experiments

In these experiments we freeze the trained encoder from previous experiments and train a classifier on top of the latent document representations to probe their capacity to reconstruct labels that reflect global semantic classes of the source documents. This is an application of a transfer learning methodology for evaluating representations. This task involves predicting the correct class label for a document based on the section in the newspaper it appears, in total 11 sections. Since there is significant class imbalance in our source data we need to establish a chance baseline. Rather than computing this using statistics we take a practical approach and use a randomized sample method. We take our training data with class labels and randomly shuffle the labels and then compute the accuracy of the randomly shuffled labels. We repeat this process 10 times to get a good estimate of chance accuracy which is 12.4%, this gives us a lower bound on the task. To obtain an upper bound we train a new task-specific model comprising of the same encoder we use in the previous sequence transduction tasks with fully trainable weights coupled with a single layer perceptron that predicts class probabilities as in the class prediction side task. This model obtains a validation accuracy of 23.7% on the class prediction task, which is our upper bound.

We had expected the upper bound to be significantly higher to give a greater range for this to be useful for a metric, however this was not the case. Table 5.10 shows the results for our different models and the upper and lower bounds for the task for reference. We can see that all models perform above chance on this task, meaning that their latent representations learn some information that is useful for reconstructing the class labels. As we hypothesized the summarization single task has the worst performance of our models on this task however the difference in performance is modest, with the TTSG single-task model, VEDA-tag, scoring just 1.16% higher accuracy. The three MTL models perform best, scoring between 20.22% and 20.60%, however with such small differences between them we cannot justify significant debate around these scores without a statistical analysis, which was beyond the scope of the present research as it would require several reruns for each of the models. Figure 5.11 shows a 2D projection of the encoder latent space representations of the validation dataset for the multitask model ($\alpha = 0.85$) coloured by the true target class of the transfer learning side task. It is clear from visual inspection that the model has learned separable representations across some, but not all, of the classes.

These findings give some weight toward our hypotheses regarding latent represen-

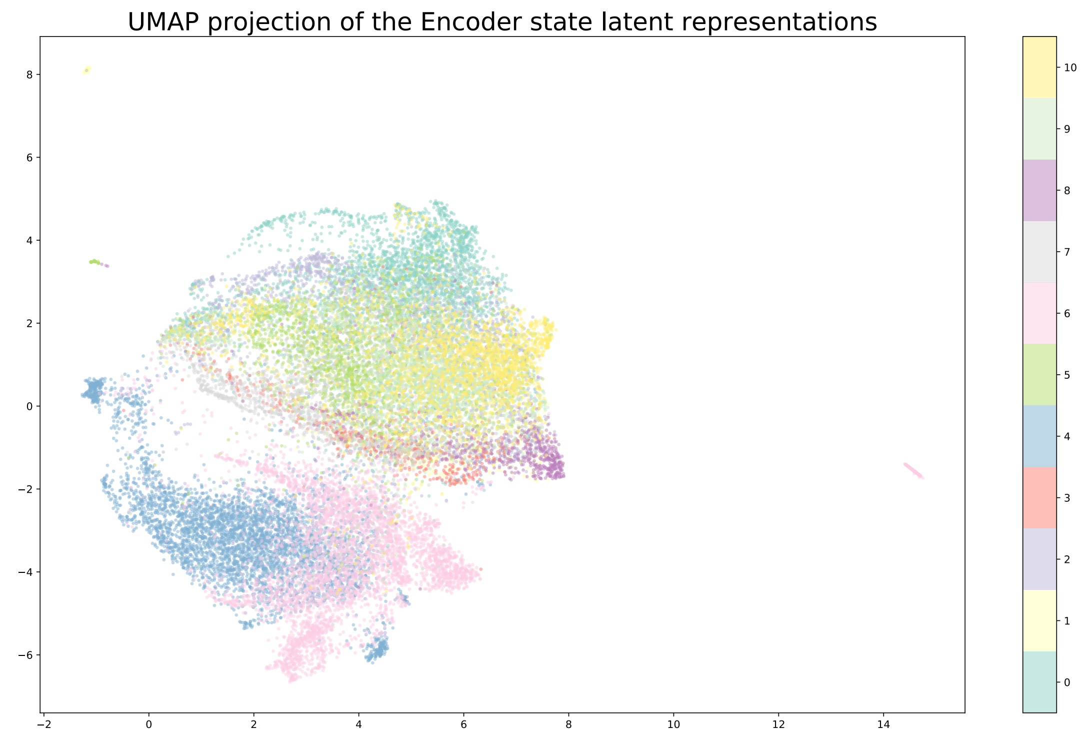

Figure 5.11: This figure shows a 2d projection of the encoder latent state representations used for transfer learning coloured by target class labels of the side task for the multitask model ($\alpha = 0.85$). We can see visually the separation of some classes from others. The projection is done using Uniform Manifold Projection and Approximation (UMAP, McInnes and Healy (2018)

tations. Firstly, that the topic tag sequence model, VEDA-tag, learns latent representations that are better at reconstructing the document semantic class than the representations learned by the single-task summarization model. Secondly, that by training a summarization model as part of a MTL model with an auxiliary tag sequence generation task we encourage learned latent representations that are also more useful for predicting the document semantic classes. We suggest that the semantics-oriented auxiliary task has increased the gathering of semantic information from the source texts. This, however, did not lead to improved summaries due to the observed catastrophic failure of the MTL models.

Model	Text
REF summary	”brexit so complex it could overwhelm politicians , warn senior academics . independent group says leaving eu will test constitution and legal framework to their limits and possibly beyond .”
REF tags	”eu_referendum_and_brexit, european_union, foreign_policy, politics, uk_news, academic_experts, business, europe, world˙news”
VEDA-summ	”theresa may to meet uk over eu membership plan . former health secretary says uk s eu membership will be a priority for britain s future , but says it will not be able to meet .”
VEDA-tag	”eu_referendum_and_brexit, uk_news, european_union, foreign_policy, politics”
MTL-summ (equal loss)	”uk : uk uk to to to to to to to to , ...”[trunc.]
MTL-tag (equal loss)	”arizona_cardinals, arizona_cardinals, arizona_cardinals, communism, communism, communism”
MTL-summ ($\alpha = 0.7$)	”eu eu eu eu eu eu eu eu eu eu eu eu eu eu eu eu eu...”[trunc.]
MTL-tag ($\alpha = 0.7$)	”art_and_design, art_and_design, art_and_design...”[trunc.]
MTL-summ ($\alpha = 0.85$)	”theresa eu eu eu eu eu eu eu eu the the the the...”[trunc.]
MTL-tag ($\alpha = 0.85$)	”eu_referendum_and_brexit, eu_referendum_and_brexit, eu_referendum_and_brexit,...”[trunc.]

Table 5.9: Example generated outputs for the multitask models. Outputs truncated where the same token is repeated.

model	Accuracy (%)
Chance (lower bound)	12.41
VEDA-summ	18.98
VEDA-tag	20.14
MTL Equal loss	20.52
MTL alpha = 0.7	20.22
MTL alpha = 0.85	20.60
Task-specific (upper bound)	23.75

Table 5.10: Results for the document global semantic class prediction side task.

Chapter 6

Conclusion

In this research we present a series of experiments investigating encoder-decoder models for sequence transduction and methods for inspecting and manipulating the learned latent representations, in particular through a MTL framework. We do this within the context of a novel dataset of news articles and associated side information from the Guardian newspaper. After describing relevant aspects of the novel dataset we train several models using recent SOTA architectures and an existing modelling library, Open-NMT, to establish high quality benchmarks. We then train our own model using a relatively simple architecture and show that its performance is equivalent to the Open-NMT model. We compare our new dataset with the existing CNN / Daily Mail benchmark dataset for abstractive summarization and show that it is a reasonable candidate task and has some appealing qualities, such as a high novel bigram rate for the reference summaries. We perform a regression analysis of the ROUGE metrics across our different models and show that high and low performance are not consistent across models and test examples. This is an interesting finding since it suggests that an ensemble of different models with a learned policy to weight outputs toward a particular model conditioned on qualities of the source text might lead to significant gains in performance.

We then show that the same encoder-decoder model can be used to learn to generate reasonably high quality sequences of topic tags, a piece of side information associated with the articles. We combine the two sequence transduction tasks using an encoder-dual-decoder model as an example of hard parameter sharing MTL. We fail to train a model able to generate high quality outputs for either task under MTL, despite trying several adaptations including a weighted compound loss function and modification to the source data. We consider several causes for this failure, including the compound loss function, limited representational capacity in the model, the particular MTL architecture used, too small training data, too few training epochs and under-regularization. We speculate that the task objectives are adversarial and lead to detrimental performance on both tasks and catastrophic failure of the model to learn good representations for either task. This finding is in contrast to the reviewed literature finding MTL to be beneficial to an number of NLP tasks (Collobert and Weston, 2008; Ruder, 2017).

Finally, we construct an experiment to probe the information captured by the latent representations learned by the encoders across our various tasks. We do this using a transfer learning approach, where we freeze the weights of the trained encoders and detach this from the decoder. We then train a classifier over the latent representations to predict the semantic class of the source document. We show that, whilst the single-task summarization model learns some information useful for reconstructing the article class, it learns less information that the encoder for the TTSG task and by combining the two tasks using the encoder-dual-decoder MTL model we encourage representations that contain more information relating to the global article semantics resulting in increased prediction accuracy for the class prediction side task.

6.1 Future Work

We identify a number of areas where further research might prove fruitful. We find evidence for qualitative differences in high and low performance for different sequence transduction architectures. Further investigation of these differences might lead to successful ensemble strategies that give rise to adaptive models conditional on a pre-analysis of the source text. Whilst a departure from the holy grail of unified architectures these may lead to performance improvements. We also identify differing performance across article categories. It is quite likely there are different ’styles’ of summarization, future work might investigate models that formally account for this latent conditional information (Some work has been done in this area already e.g. Narayan (2018)).

Our research would have been improved significantly with the inclusion of human evaluation. Although platforms like Amazon’s Mechanical Turk have made this wildly more accessible there is still a significant barrier in including human evaluation in an agile way. Better automated methods would also be beneficial (Novikova et al., 2017). During the course of this research we made some early attempts at using universal sentence embeddings and simple distance metrics[1] (Conneau et al., 2017). Interestingly, sentence embeddings may solve two issues in automatic evaluation simultaneously. Firstly, they allocate importance weights to individual words in a sentence, based on training on a very large corpus (see figure 6.1), meaning we can create weighted evaluations that ignore less important words. Secondly, by computing scores based on embedding distances we are no longer restricted to evaluation based on appearance of exact tokens, since a high quality paraphrase using a different vocabulary will still appear close in this evaluation space. By transforming the reference and generated sequences to sentence embeddings and computing, for example, the cosine similarity of the resulting vectors we get a distance measure that accounts for word importance and is sympathetic to synonym exchange. We were able to implement this however we were unable to perform a robust evaluation. Anecdotally it appeared to perform as well as, or better, than ROUGE in some cases. A limitation is that this metric is less interpretable than ROUGE, so it could not serve as a replacement. It could also, alternatively, be used as a value function during training (or as a loss function in the form $1 - distance$). Further research into such approaches might prove valuable.

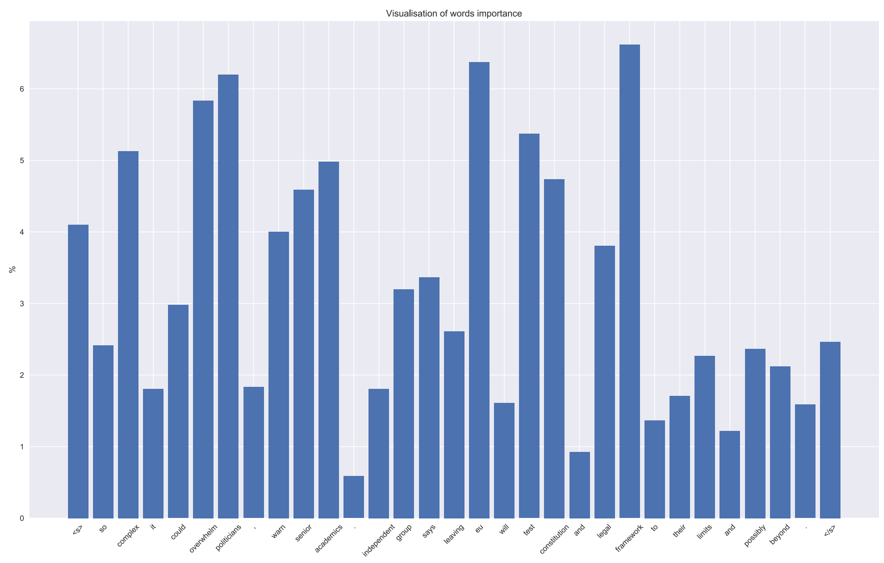

Figure 6.1: Word importance weights for a reference summary from our dataset.

We would also recommend exploring these approaches with larger and more varied datasets, possibly by combining data from different summarization tasks to see if this

[1] not reported as this work was not completed but we consider it work discussing as future work.

leads to more generalizable summarization strategies. The present work would also be improved upon by considering better regularization strategies such as variational dropout (Gal and Ghahramani, 2016) and zoneout (Krueger et al., 2016). We feel that the loss function used may be hindering the learning dynamics and that there might be better loss functions leading to better sequence outputs, particularly in the case of the summarization task. The MTL model we explored is only one of many approaches that could be taken, we would also like to consider soft parameter sharing methods, and other MTL structures such as cascaded or triangular MTL (Anastasopoulos and Chiang, 2018).

Alternative attention functions may also enable higher performance on these types of tasks. For example multi-head attention enables the model to jointly attend to information from different representation subspaces at different positions (Vaswani et al., 2017). Other model architectures such as the Transformer (Vaswani et al., 2017) or Convolutional sequence-to-sequence (Gehring et al., 2017) models might also perform better on these tasks.

We also consider the suitability of our auxiliary task. We assumed that our side task of topic tag sequence generation was complimentary to the primary summarization task, however perhaps this is not the case. Although previous research has made use of dual and multi-decoder models (Luong et al., 2015b) our work is the first to our knowledge where the two tasks are not monotonically aligned. We may see greater success, for example, if we reordered the tag sequences to align better with the reference summaries.

Appendix A

Appendix

Model	Text
SOURCE	”it was the night that london s underground finally became a 00-hour service , or , as one driver put it in an announcement : it s 0am and you lot are still on the tube . the victoria and central lines are the first two underground lines launching a night service . the northern , jubilee and piccadilly lines are expected to follow suit before the year is out . friday s service started with little fanfare at walthamstow central , as the 00.00 departure , newly classified on the timetable as a night tube , left with only a few people on board . naso koutzoukis was one of them . originally from athens , and having lived in london for five years , he d travelled on the train specifically to head in to town to see the drunken crowds . it should be fun . naso koutzoukis travelling in the first night tube train on victoria line . related : londons night tube set for weekend debut passenger numbers swelled and the volume of chat rose as the train headed into central london . the aptly named victoria from brixton was unconvinced it really counted as the victoria line night tube yet , as a train ran at this time every evening . she began asking people how they felt about being on the almost night tube . i m a f**king pest on the tube . people either love it or hate it , but i m always asking who are you , where are you going , what are you doing . victoria from brixton in full pest mode , talking to fellow passengers . on the return journey from brixton , kevin chauphary was much more sure he was on an actual night tube . i definitely think it s a momentous occasion for all of us londoners . it increases safety massively . you are not just loitering around waiting for a night bus . london s night tube service kicks off night bus services have also been improved to coincide with the night tube s launch , with eight routes extended to 00-hour operation to provide additional connections to areas being served by the night tube . at 0.00am , back at walthamstow central , with a flurry of tfl officials , press people and a couple of guardian angels in...”

Table A.1: Source text for generated summaries in table 5.2 in chapter 5.

Model	Text
SOURCE	”crystal palace are set to break their club record fee to sign the wigan midfielder james mcarthur for 0m . mcarthur s transfer would surpass the former club record fee paid to peterborough last summer for dwight gayle , and would cost palace the same figure wigan agreed with leicester city before they opted instead to secure esteban cambiasso on a 00-month deal . there had been suggestions over the weekend that palace might send adl ne guedioura the other way , with the finer details of the deal still to be finalised . talks continue with tottenham hotspur over zeki fryers , whom palace had initially hoped to secure on loan but may now have to sign permanently . the young left back is apparently already at the training ground in beckenham in anticipation of the completion of that move to south london . glenn murray could also leave selhurst park but only if the new manager neil warnock can secure another striker as a replacement . nottingham forest , reading and ipswich are all thought to be interested in signing the player who scored 00 championship goals in palace s promotion season in 0000-00.”

Table A.2: Source text for generated summaries in table 5.3 in chapter 5.

Bibliography

Anastasopoulos, A. and Chiang, D. (2018). Tied multitask learning for neural speech translation. *CoRR*, abs/1802.06655.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 17–36.

Bengio, Y., Courville, A. C., and Vincent, P. (2012). Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Britz, D. (2016). Deep learning for chatbots, part 1 introduction. Article on wildML.com, at http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/.

Britz, D., Goldie, A., Luong, M., and Le, Q. V. (2017). Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th*

International Conference on Machine Learning, ICML ’08, pages 160–167, New York, NY, USA. ACM.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364.

Despois, J. (2017). Latent space visualizationŁŁdeep learning bits #2. Article on Medium.com, at https://hackernoon.com/latent-space-visualization-deep-learning-bits-2-bd09a46920df.

Gal, Y. and Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122.

Ghader, H. and Monz, C. (2017). What does attention in neural machine translation pay attention to?

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Guo, H., Pasunuru, R., and Bansal, M. (2018). Soft layer-specific multi-task summarization with entailment and question generation. *CoRR*, abs/1805.11004.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359 – 366.

Howard, J. and Ruder, S. (2018). Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.

Inan, H., Khosravi, K., and Socher, R. (2016). Tying word vectors and word classifiers: A loss framework for language modeling. *CoRR*, abs/1611.01462.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F. B., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2016). Google’s multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558.

Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). Opennmt: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Krueger, D., Maharaj, T., Kramár, J., Pezeshki, M., Ballas, N., Ke, N. R., Goyal, A., Bengio, Y., Larochelle, H., Courville, A. C., and Pal, C. (2016). Zoneout: Regularizing rnns by randomly preserving hidden activations. *CoRR*, abs/1606.01305.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Liu, X., Gao, J., He, X., Deng, L., Duh, K., and Wang, Y.-Y. (2015). Representation learning using multi-task deep neural networks for semantic classification and information retrieval. NAACL.

Lo, H. Y.-C. (2018). Pytorch seq2seq example. Tutorial on github.io, at https://github.com/howardyclo/pytorch-seq2seq-example.

Luan, Y., Brockett, C., Dolan, B., Gao, J., and Galley, M. (2017). Multi-task learning for speaker-role adaptation in neural conversation models. *CoRR*, abs/1710.07388.

Luong, M., Pham, H., and Manning, C. D. (2015a). Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025.

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2015b). Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

Luong, T., Socher, R., and Manning, C. (2013). Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.

McCann, B., Shirish Keskar, N., Xiong, C., and Socher, R. (2018). The Natural Language Decathlon: Multitask Learning as Question Answering. *ArXiv e-prints*.

McInnes, L. and Healy, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Nallapati, R., Xiang, B., and Zhou, B. (2016). Sequence-to-sequence rnns for text summarization. *CoRR*, abs/1602.06023.

Narayan, S. (2018). Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. Accepted to appear at EMNLP 2018.

Narayan, S., Papasarantopoulos, N., Lapata, M., and Cohen, S. B. (2017). Neural extractive summarization with side information. *CoRR*, abs/1704.04530.

Novikova, J., Dusek, O., Curry, A. C., and Rieser, V. (2017). Why we need new evaluation metrics for NLG. *CoRR*, abs/1707.06875.

Olah, C. (2017). Understanding lstm networks. Article on github.io, at http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Pan, S. J., Yang, Q., et al. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Panchapagesan, S., Sun, M., Khare, A., Matsoukas, S., Mandal, A., Hoffmeister, B., and Vitaladevuni, S. (2016). Multi-task learning and weighted cross-entropy for dnn-based keyword spotting. In *INTERSPEECH*, pages 760–764.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Pasunuru, R., Guo, H., and Bansal, M. (2017). Towards improving abstractive summarization via entailment generation. In *NFiS@EMNLP*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Press, O. and Wolf, L. (2016). Using the output embedding to improve language models. *CoRR*, abs/1608.05859.

Renals, S. (2017). Machine learning practical lectures. University of Edinburgh course. Website available at http://www.inf.ed.ac.uk/teaching/courses/mlp/.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533 EP –.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Shwartz-Ziv, R. and Tishby, N. (2017). Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810.

Sundermeyer, M., Alkhouli, T., Wuebker, J., and Ney, H. (2014). Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14–25.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

Tardy, P. (2017). A full python librarie for the rouge metric. Github repository on github.io, at https://github.com/pltrdy/rouge.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.